# Optimization-based inverse model of soft robots, with contact handling

Eulalie Coevoet

Université Lille 1

Eulalie Coevoet

# Optimization-based inverse model of soft robots, with contact handling.

Thesis submitted in fulfillment of the requirements for
the degree of Doctor of Philosophy in Computer Science.

**Committee:**

| | |
|---|---|
| François Chaumette, *Research Director at INRIA Rennes* | Chair |
| Jamie Paik, *Assistant Professor at EPFL* | Reviewer |
| Nicolas Mansard, *Permanent Researcher at LAAS/CNRS Toulouse* | Reviewer |
| Allison Okamura, *Professor at Standford University* | Examiner |
| Hadrien Courtecuisse, *Permanent Researcher at CNRS Strasbourg* | Examiner |
| Adrien Escande, *Permanent Researcher at CNRS-AIST JRL* | Examiner |
| Christian Duriez, *Research Director at INRIA Lille* | Supervisor |

**Thèse soutenue le 9 janvier 2019**

# Abstract

Soft robotics draws its inspiration from nature, from the way living organisms move and adapt their shape to their environment. In opposition to traditional rigid robots, soft robots are built from highly compliant materials, allowing them to accomplish tasks with more flexibility and adaptability. They are safer when working in fragile environment. They have the advantages of producing low forces that are suitable for manipulating/interacting with sensitive objects/surroundings without harming them. These characteristics allow for potential use of soft robotics in the fields of manufacturing and medicine.

But the field of soft robotics brings new challenges, in particular for modeling and control. Within this thesis we aim at providing generic methods for soft robot modeling, without assumptions on the geometry. The methods are based on the finite element method to capture the deformations of the robot's structure and of its environment when deformable. We formulate the problem of their inverse kinematics and dynamics as optimization programs, allowing easy handling of constraints on actuation and singularity problems. We are able to control several types of actuation, such as cable, pneumatic and hydraulic actuations.

Moreover, most of the applications involve interaction of the robot with obstacles. Yet soft robots kinematics is highly dependent on environmental factors. We propose new methods that include contacts into the optimization process. These methods make an important step as we think that the knowledge of contacts in the modeling is all the more important. Finally, we propose to control some soft robots during locomotion and grasping tasks which require the use of contact with static friction. We give a particular attention to provide solutions with real-time performance, allowing online control in evolving environments.

# Glossary

**DoF** Degree of Freedom 12

**DoFs** Degrees of Freedom 12, 20

**FE** Finite Element 12

**FEM** Finite Element Method 11, 23, 35, 45

**ID** Inverse Dynamics 19, 31

**IK** Inverse Kinematics 19, 31

**IP** Inverse Problem 19, 20, 43, 139

**LCP** Linear Complementarity Problem 96, 99, 117

**MCP** Mixed Complementarity Problem 97

**NLCP** Non-Linear Complementarity Problem 117, 118

**QP** Quadratic Program 27, 37, 45, 68, 135

**QPCC** Quadratic Program with Complementarity Constraint 27, 44, 45, 97, 98, 112, 118, 134

**SOFA** Simulation Open Framework Architecture 11, 25, 71

# Contents

Chapter 1

# Overall Introduction

## Contents

## 1.1 Introduction

The context of the thesis is wide, as it involves soft robotics and numerical simulation. We give this first chapter to an overall introduction of this context. Note that the state of the art related to the thesis contributions (i.e. control of soft robots) will be discussed in chapter 2.

We start this chapter with a section dedicated to the soft robotic field 1.2. We first set the definitions that give to a soft robot its characteristics and properties. We then briefly present the motivations around soft robotics, talk about their design and the extent of possibilities handled by our methods. We finally end the section by introducing the challenge we are interested in, that is their modeling and motion control. The methods we propose are based on a numerical representation of the robot using the Finite Element Method (FEM). The third section of this chapter is thus dedicated to numerical simulation 1.3, as it is the foundation of our algorithms. We start this third section by briefly introducing the FEM, and in particular the benefits and limits of the method. We then discuss the possibilities in terms of simulation software, and present the Simulation Open Framework Architecture (SOFA) that we use for this thesis. We finally give an end to this overall introduction, by stating the thesis contributions 1.4, and giving the manuscript outlines 1.5.

## 1.2 Soft robotics

Robots are primarily known for the way they revolutionized the manufacturing sector, by replacing mankind with difficult and repetitive tasks. They are also revolutionizing the medical sector, where they are now assisting surgeon in operating room with minimally invasive surgical procedures. All these robotic systems are traditionally made from stiff material. Yet, one issue of robotics is to remain safe for interactions with fragile environments. Indeed, for instance, it is of interest to have robots able to manipulate fragile items in the manufacturing sector, to work with human in a collaborative way without harming them, or having surgical robots that can interact safely with human organs. This issue has been the driving factor of soft robotics emergence, as using soft materials could clearly improve the safeness.

The first so called soft robot is maybe the one proposed by Suzumori. First, in (Suzumori et al. 1991), Suzumori et al. proposed a soft micro-actuator made of fiber-reinforced rubber and driven by pressure with a very simple mechanism. This work was inspired by the need of medical micro-robots for inspecting the body. In (Suzumori 1996), Suzumori applies these actuators as robot fingers, creating soft robots able to grasp objects, or even screw bolts (see Figure 1.1).

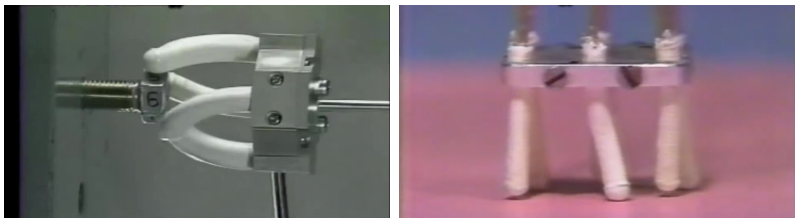

Figure 1.1: *Soft robots from (Suzumori 1996). (left) Soft robot ables to screw a bolt. (right) Small-scale soft robot ables to walk.*

They also created small-scale robots able to walk. This work was pioneer, and it took ten years for the field of soft robotics to emerge and become active.

We identify distinct classes of research within the soft robotics field: design, fabrication, modeling and control. The most active ones are maybe design and fabrication, while we are interested in the modeling and control part. First let us give some definitions.

### 1.2.1 Definitions and properties

Some terminologies classify the robots, such as redundant, continuum, hard and soft. In this section we provide definitions of these terms widely used in the robotics community. These definitions, that give the characteristics and properties of soft robots, are key elements to understand the problematic of their control:

*Soft Robots:* Class of robot made of soft materials, usually with Young moduli between $10^4$ and $10^9 Pa$ (Majidi 2014), such as silicone rubber and soft plastic.

*Hard Robots:* This refers to the class of robot made of hard materials, usually with Young moduli greater that $10^9 Pa$ (Majidi 2014), such as metals and hard plastics. We may also call them *Rigid* robots.

*Hybrid Robots:* Class of robot made of both hard and soft materials.

*Continuum Robot:* A continuum robot is a robot capable of continuous deformations. Note that all soft robots are capable of continuous deformations, but not all continuum robots are soft. Robots made of certain hard materials such as shape memory alloys can also be designed to have continuous deformations (Trivedi et al. 2008). Note that here we consider that a continuum robot should deform its structure to behave.

*DoFs:* Degrees of Freedom (DoFs) designate the number of possible independent relative motions of a robot. For example, traditional manipulators made of rigid links connected by joints have a finite number of DoFs, generally up to six (three rotations, three translations), while continuum robots have theoretically an infinite number of DoFs; the way they continuously bend their structure leads to an infinite number of possible independent relative motions. Note that the notion of Degree of Freedom (DoF) have quite the same meaning in FEM: we discretize continuous structures with large but finite number of Finite Element (FE) DoFs, referring to the vertices of the FE mesh, where models can deform.

*Discrete:* Said of a robot with a finite number of DoFs.

*Redundant:* As kinematically redundant. Kinematic redundancy occurs when a robotic system has more DoFs than the end-effectors. In particular, when the DoFs space is greater than the task space (Siciliano 1990). With rigid robots, the redundancy is a relative concept, as it holds with respect to a given task, while all continuum robots are redundant. Note that with soft robotics, these extra DoFs can be used to follow obstacles surface, or to avoid contact (when these extra DoFs are controllable).

*Hyper-redundant:* The word hyper-redundant is used to denote robot that have a large or infinite degree of kinematic redundancy (Burdick & Chirikjian 1998), which is the case with continuum robots.

*Underactuated:* A robot is said to be underactuated if it does not have an actuator for each of its DoFs, which is the case for continuum robots since their structures possess infinite number of DOFs.

To summarize, soft robots have an infinite number of DoFs, they are hyper-redundant, underactuated, capable of continuous deformations, and as defined in (Trivedi et al. 2008), are classified on the basis of the compliance of their underlying materials. The reader can refer to the scheme 1.2 for an illustration of these characteristics inclusions, and example of robots falling in each category.

The methods proposed in this thesis have only been tested on robots entirely made of soft material, but there is no apparent limitation on applying these methods to hybrid robots. We use a simulation framework that handles the modeling of both hard and soft materials, and that allows to simulate hybrid structures. Such design should make no difference in our process of finding the inverse of the robot model, neither for the management of contacts.

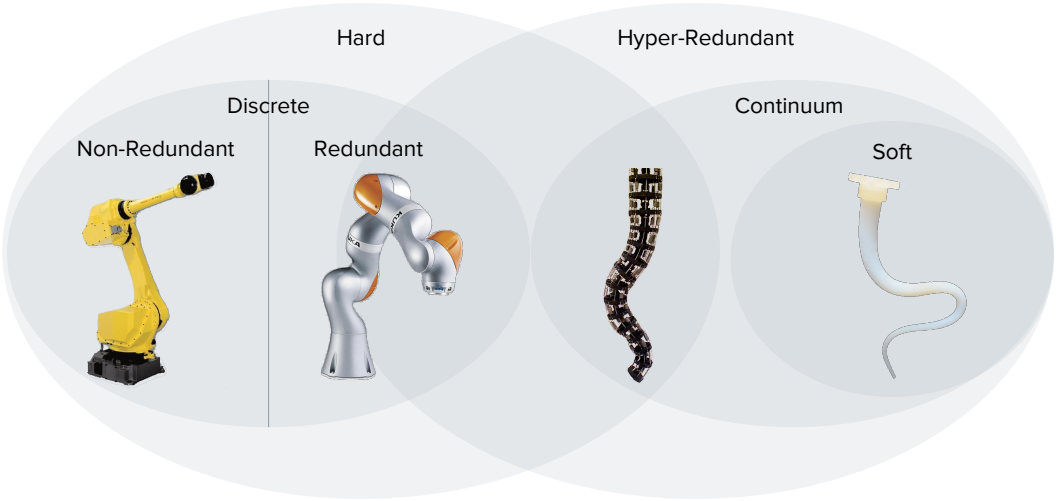


Figure 1.2: *Image and description from (Thuruthel et al. 2018). Evolution of rigid-link manipulators based on discrete mechanisms to bioinspired continuum robotic manipulators based on structures capable of continuous bending, studied in detail in (Trivedi et al. 2008).*

### 1.2.2 Bio-inspiration and motivations

Today's soft robots are particularly inspired by living organisms we can find in nature, such as snakes, caterpillars, octopi's tentacle, or elephants' trunk (see Figure 1.3). These organisms have a continuous, highly deformable structure, and yet demonstrate inspiring complicated motions. Nature made them evolved over millions years to achieve certain behaviors. Using this natural optimization to developed robots with new capabilities is needless to say of great interest.

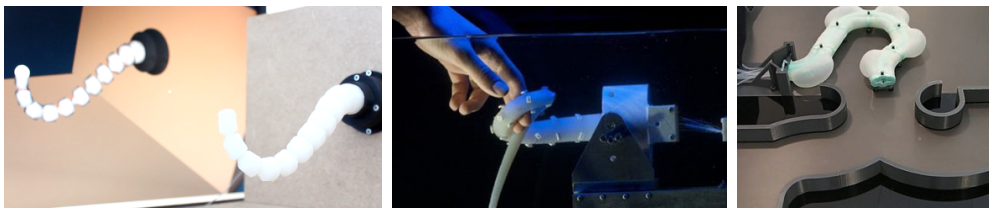

Figure 1.3: *(left) Our elephant's trunk with its simulated FE model. (middle) Octopus' tentacle from the BioRobotics Institute of the Scuola Superiore Sant'Anna. (right) Snake like soft robot from the MIT's CSAIL lab.*

If we had to present just one example of bio-inspiration, it would be the recent work of Rafsanjani et al. on a soft robot locomotion based on crawling (Rafsanjani et al. 2018). They drew their inspiration from the friction-assisted locomotion of snakes, and introduced a flexible skin with anisotropic frictional

properties (with the use of Kirigami) that enables a single soft actuator to propel itself (see Figure 1.4). This work is really nice and promising. At present, their robot is limited as it can only move forward. To enable the robot to bend and change direction, it would require the use of additional actuators.

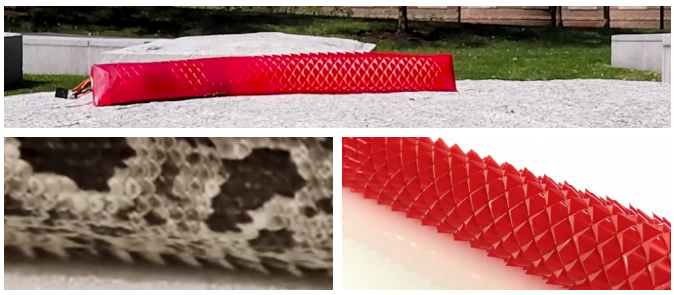

Figure 1.4: *Snakeskin robot from the Harvard John A. Paulson SEAS (Rafsanjani et al. 2018).*

Soft robotics is thus an opportunity to provide robots new capabilities, and to re-thinking the way they are designed and used. These robots are usually cheap and more easy to fabricate than rigid robot, and yet still capable of complex motions. We already mentioned that their compliance decreases risks of damage for both the robot and its surrounding, but also, using large strain deformation, they can reduce or extend their nominal dimensions, bend, and adapt their shape to the environment. These characteristics make them particularly suitable for exploration and manipulation in congested and sensitive environments.

Soft robots are indeed mainly design for grasping and locomotion. Hawkes et al. are for example looking at soft robots that navigate their environment through growth (inspired by plants), for rescue applications (Hawkes et al. 2017). Cianchetti et al. propose to use soft robots for locomotion and grasping under water (Cianchetti et al. 2015). They can also be used for other tasks. For example, Roche et al. investigate the use of soft robotics to help hearts beat, potentially opening new treatment options for people suffering from heart failure (Roche et al. 2017).

For this thesis, we designed and built few soft robots with very simple behaviors. One of them, shown in Figure 1.3, was inspired by the elephants' trunk. Its design is presented in chapter 3. We mainly designed and built these robots to demonstrate the feasibility of the methods we propose. Note that, we also confront our controllers with simulated version of some soft robots proposed by other labs.

### 1.2.3 Design and fabrication

The use of soft materials greatly challenges researchers with design and fabrication. They have to make use of new techniques to make the structure ables to deform itself, and also rethink the actuation like Rafsanjani et al. did.

#### Material

Soft robots can be built from silicone rubbers like in (Shepherd et al. 2011), soft plastics (e.g. grippers from Soft Robotics Inc), and also fabrics like in (Yuen et al. 2014). Several works like (Zehnder et al. 2017) also propose to use composite silicone to exhibit desired macroscopic mechanical properties (see Figure 1.5 (top-left)). Silicone rubber is often used when targeting high compliance. The usual process of fabrication consists in casting the silicone into a 3D printed mold. An interesting work proposes a computational approach to design flexible 3D printed molds automatically (Malomo et al. 2016), and ease the process. It becomes more complex when dealing with multiple rigidities or with cavities: several parts may need to be casted separately and then recombined (like illustrated in Figure 1.5 (top-right)). In such case the casting may introduce small asymmetries.

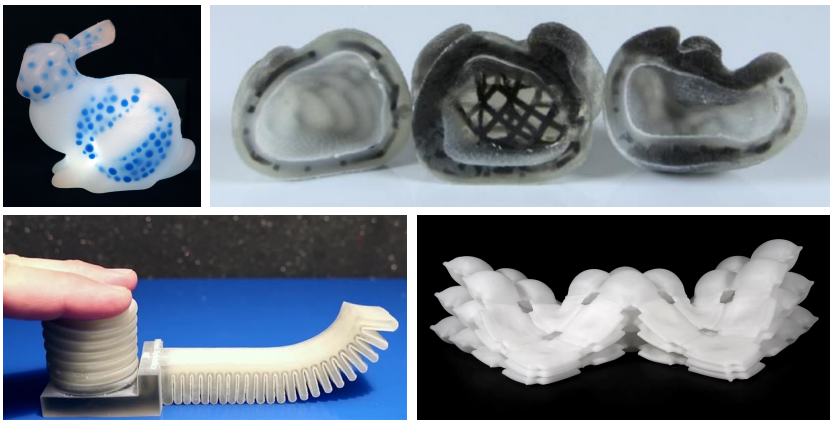

Figure 1.5: *(top-left) MetaSilicone from (Zehnder et al. 2017). (top-right) 3D printed pneumatic objects with desired deformations from (Ma et al. 2017). (bottom-left) Printable hydraulics from (MacCurdy et al. 2016). (bottom-right) Liquid printed inflatable from BMW and MIT.*

Advances in additive manufacturing enable direct 3D printing of flexible materials. Note that these materials are not as soft as some silicone rubber. Here is a nice review on 3D printing for soft robotics system (Wallin et al. 2018). In (Ma et al. 2017), Ma et al. propose to optimize chamber structures and material distribution inside the object to reach target deformations (see Figure 1.5

(top-right)). Another great work has been made on fully 3D printed soft robots actuated with liquids (MacCurdy et al. 2016). What is interesting in this work, is that they print liquids together with solids (hard and soft), and thus print one part actionable objects (see Figure 1.5 (bottom-left)).

An astonishing work on liquid printed deformable and inflatable objects has been recently proposed by BMW Design and MIT's Self-Assembly Lab (see Figure 1.5 (bottom-right) and weblink). Based on the new 3D-printing technology proposed by the MIT, they are able to directly print silicone within a gel suspension. This technique enables to print highly inflatable object without dealing with cavity support removal.

Interesting work, from the computer graphics community, also look at the use of micro-structures to reach macroscopic target elasticity. For example, in (Schumacher et al. 2015, Panetta et al. 2015) authors proposed to use assembly of elementary material tiles, and (Martínez et al. 2016) proposed to use micro-structures inspired by Voronoi open-cell foams, each of them to fabricate objects with spatially varying elasticity. (Ion et al. 2016) and (Zhu et al. 2017) go a bit further by proposing to use micro-structures (multi-material microstructures for Zhu et al.) to perform mechanical tasks. It consists of building a single block of material, with defined and arranged cells, which together achieve desired macroscopic movement. See Figure 1.6. We think these works could be interesting for the design of soft robots. It could for example help at reducing the number of actuators, by using passive structural behaviors. Note that high compliance can be reached with some of these approaches.

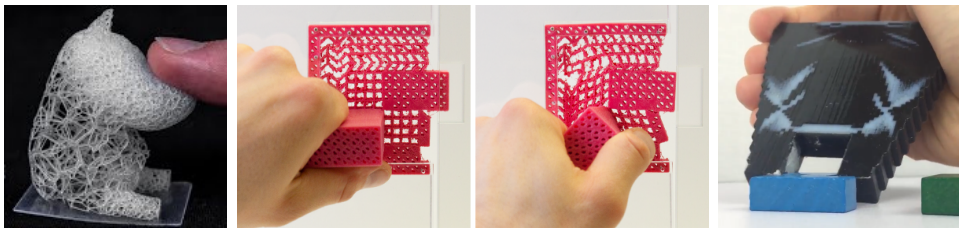

Figure 1.6: *(left) Procedural Voronoi foams from (Martínez et al. 2016). (middle) Meta-material mechanism from (Ion et al. 2016). (right) Multi-material microstructures mechanism from (Zhu et al. 2017)*

As already mentioned, the methods we propose are based on a numerical representation and simulation of the soft structures. Although we will discuss numerical simulation in section 1.3, we would like to already give few comment on the modeling of these different materials. From a simulation point of view, plain structures, made of silicones or soft plastics for example, are easy to

model. Structures like meta-silicone are a bit more complex to handle: they require taking into account the materials boundaries into the FE mesh, i.e. they require having points on these boundaries to be able to isolate elements corresponding to the different materials. The simulation framework we use, proposes tools, based on the Computational Geometry Algorithms Library (CGAL), to generate such meshes. With a distribution of different Young's moduli over the structure it is then possible to simulate composite silicone. Other techniques can also be used, like coarse embeddings proposed in (Nesme et al. 2009). Note that this simulation framework also enable the simulation of anisotropic material.

Micro-structures have been approximated with the use of homogenization theory (Cioranescu & Donato 1999). This theory consists in simplifying the modeling and simulation of micro-structures, by averaging the microscopic behavior, and use a coarser macroscopic discretization with an equivalent behavior at the macroscopic scale. Indeed, it is not conceivable to mesh with precision and simulate the complexity of micro-structures. Although we do not currently consider such structures in the thesis, nor anisotropic behavior, our inverse methods should be directly applicable.

**Actuation**

Researchers have categorized actuation as either intrinsic or extrinsic, depending on the location of actuation: "within the moving manipulator structure itself (intrinsic) or outside of the main structure with forces transmitted to the structure through some mechanical transmission (extrinsic)." (Burgner-Kahrs et al. 2015). Here are some types of actuation used with soft and hybrid robots:

- Extrinsic: cable actuation (Vikas et al. 2016), micro motor directly link to the soft structure, shape memory alloy (SMA) as external actuation (Seok et al. 2013).
- Intrinsic: pneumatic actuation (Terryn et al. 2017), hydraulic actuation (MacCurdy et al. 2016), vacuum actuation (Li et al. 2017), jamming as a mechanism that modulate actuators output (Steltz et al. 2009), McKibben muscles (Roche et al. 2014), internal combustion (Tolley et al. 2014).

Each actuation technique having advantages and drawbacks. For example, the use of cable requires to deal with frictional effect between the wire and the structure, but allows large deformations, with lower stress on the structure than pneumatic actuation for instance. An interesting work is using self-healing

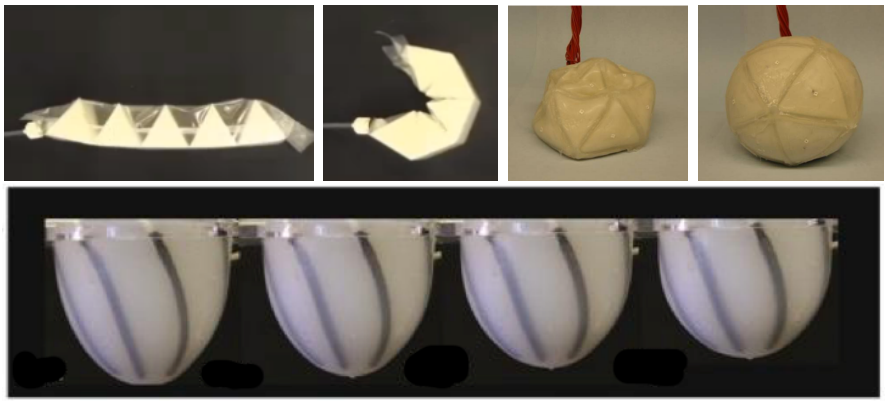

Figure 1.7: *(top-left) Vacuum actuation from (Li et al. 2017). (top-right) Jamming mechanism from (Steltz et al. 2009). (bottom) McKibben muscles from (Roche et al. 2014).*

elastomer (Terryn et al. 2017) to get round of material breaks and leaks that are commonly experienced with pneumatic actuation, due to excessive pressure and stress over the soft structure. Vacuum actuation instead of positive pressure is also a method that is investigated to avoid material break (Robertson & Paik 2017).

SMAs are small in size and yet allow to perform large deformation, but it is often at a cost of slow motion. Some actuations can also be limited (depending on the applications) by the hardness and size of the hardware, such as air compressors and servomotors. Indeed, applications involving locomotion for example may require the robots to be autonomous, and thus to have the hardware embedded into the soft structure. Like for example the robots proposed in (Rafsanjani et al. 2018), (Tolley et al. 2014), and (Vikas et al. 2016). Note that it brings out a problematic: how to preserve desired softness properties, and how to deliver desired behaviors, while embedding the hardware.

However, we think the most used actuations are cable and pneumatic actuations. The methods we propose in this thesis currently handle robots actuated with cable, pneumatic, hydraulic, and/or variation of structures Young Moduli. More simply, we also handle soft robots with parts directly attached to actuators with few DoFs (such as servomotors).

Vacuum actuation is more challenging. Although we are able to simulate and control (in the sense of finding the inverse of the model) negative pressure, vacuum actuation rests on the use of self collisions over very large surfaces. Dealing with such huge amount of contacts can be too expensive from a computational point of view.

### 1.2.4 Modeling and control

To control the motion of robots, we have to solve an Inverse Problem (IP), that is the Inverse Kinematics (IK) or the Inverse Dynamics (ID) of the robot. The term IP is common in science, whether in mathematics, physics or mechanics. It is the opposite of what we call a *forward* or a *direct* problem. As we may use the word *direct* for other purposes, we choose in this manuscript the *forward* designation. Given a function $f$, the forward problem consists in computing from a parameter $\theta$ the result $x = f(\theta)$. This can be seen as the natural way. The corresponding IP is for its part, compute the parameter $\theta = f^{-1}(x)$ from a target result $x$, if possible. For example, in our case, we aim at solving the IP of finding the actuation (a cable displacement for instance) that leads a soft robot into a desired posture (see Figure 1.8).

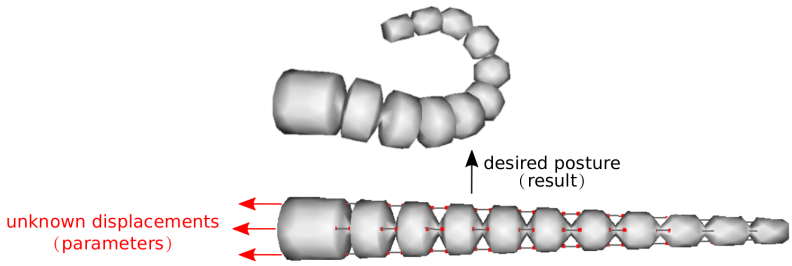


Figure 1.8: *An IP involving the motion control of a cable-driven soft trunk.*

#### Motion control

Note that IK and ID respectively corresponds to an IP based on, the kinematic description of the robot motion and the dynamic description. Kinematics describes the rotational and translational motion of bodies, without considering the forces that cause the motion nor the mass, while dynamics does. The methods propose in this thesis can be used with either description. Having said that, several difficulties are experienced in the modeling and control of soft robots. As already mentioned, soft robots have an infinite number of DoFs, they are hyper-redundant, underactuated, and capable of continuous deformations, yielding to the following difficulties:

1. In opposition to articulated rigid structures, it is hard to get an analytical model that describes the mechanical behavior with accuracy.
2. Infinite possible postures can lead to the same position of the end effector. Therefore the solution to their IK is generally non-unique.

3. Many of their DoFs are not directly controllable.

4. Their kinematics do not only depend on their geometry, many *new* parameters such as material properties, have a direct influence on their posture (see Figure 1.9). In particular, unlike rigid robots, their kinematics is highly dependent on environmental factors, such as gravity and contact with obstacles (See Figure 1.10).

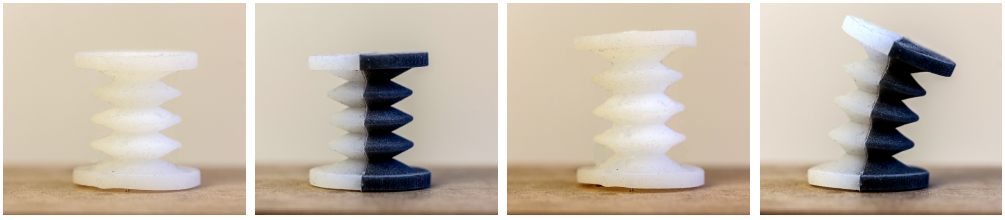

Figure 1.9: *Spring-shape robot actuated with pressured air (left and middle-left: no actuation, right and middle-right: pressure actuation). With the same structural geometry, the kinematic relationship (output deformation due to input cavity volume) changes due to the stiffness difference in the material used for the construction of the structure. Left and middle-right: the robot is made of a single type of silicone Middle-left and right: the robot is made of two types of silicone with the black silicone being stiffer than the white one.*

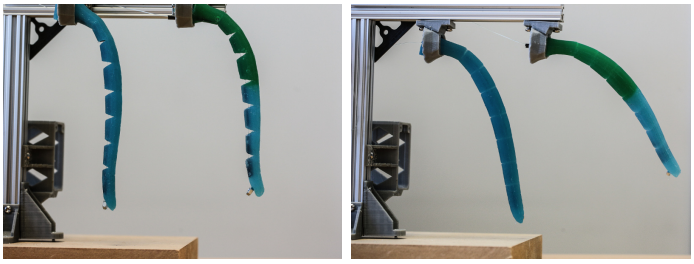

Figure 1.10: *Tentacle robots with same geometry, that can be actuated with one cable. Without actuation and under gravity, the material properties as well as geometric configuration play a key role in the resulting deformation. The blue tentacle is made of a single silicone, while the blue and green tentacle is made with two different silicones, the green one being stiffer than the blue one. While the left image illustrates that the resulting deformation between the two tentacles is already a bit different, the right image exhibits that a different orientation of the tentacle leads to even larger different deformation between the homogeneous tentacle and the composite one.*

As stated in many reviews (Trivedi et al. 2008, Kim et al. 2013, Rus & Tolley 2015), all these differences between traditional robotics and soft robotics have

an impact on the modeling tools and controllers generally in use in robotics. Traditional controllers cannot be directly applied on soft robots, and new IK and ID solutions have to be developed to control the motion of such flexible structures. Some solutions have been proposed by the community, such as controller based on analytical solutions but with strong geometric assumptions, machine learning, Jacobian methods and optimization-based methods. We discuss all these solutions in chapter 2.

### Sensing

When controlling the motion of a real robot, the use of sensors is necessary to enable a feedback closed-loop control; that is using feedback on the real robot shape to correct the model uncertainties (Zhang et al. 2016), or to feed algorithms from control theory and stabilize the process (avoid vibrations) (Santina et al. 2018). Note that sensors can also be used to calibrate the robot, when the robot parameters such as stiffnesses do not match precisely the reality, or to initialize cables length for instance. Depending on the application, sensors may need to be embedded into the soft structure, or external such as depth-cameras or motion capture systems. For example, Zhang et al. propose a feedback closed-loop control strategy based on visual-servoing in (Zhang et al. 2017). For applications where depth-cameras are not suitable, embedded sensors are required. Such sensors need to be flexible as proposed in (Vogt et al. 2013), or even stretchable as proposed in (Muth et al. 2014, Atalay et al. 2017), to leave the behaviors of the robot unchanged. See Figure 1.11.

Figure 1.11: *(left) Force sensor using embedded micro-fluidic channels from (Vogt et al. 2013). (middle) Silicone textile composite capacitive strain sensor from (Atalay et al. 2017). (right) Defsense, deformable input devices from (Bächer et al. 2016)*

Soft structures with embedded soft sensors are also proposed within the human-computer interaction community. In (Bächer et al. 2016), Bächer et al. proposed an optimization-based algorithm for the design and fabrication of deformable input devices. These devices are capable of continuously sensing their deformation, which could be very interesting for soft robotics closed-loop control. See Figure 1.11 (right).

In this thesis, all the experiments have been conducted with an opened-loop control. The results we obtained are already correct, even though it would improve them to use a closed-loop control strategy, with the use of sensors. We think a closed-loop strategy can be applied to the current work, without profound changes of the algorithms.

## 1.3 Numerical simulation

Numerical simulation has been used in various domains, including robotics and computer graphics. Depending on the application, a simulation can intend to be as accurate as possible (e.g. for medical applications), or just close enough to a physical behavior for esthetic's purposes (e.g. films and games). The methods we proposed are based on an accurate description of the soft structures using FEM, which has advantages and drawbacks. In this section we briefly introduce the FEM, and talk about the existing simulation softwares. In particular, we present SOFA, the simulation framework we used to develop our methods.

### 1.3.1 Finite element method

FEM, also called finite element analysis or finite element modeling, is used in the current work to analyze the continuous deformations of soft structures. FE method is one of the most popular methods in computational sciences. It is used to solve a wide variety of problems, such as elastic problems, fluid flow, and thermal problems to name just a few. See (Cook et al. 2007) or (Reddy 1993) for a detailed presentation of FEM. Using FEM can have numerous advantages, but also some limitations that we discuss below.

#### Benefits

Having an accurate representation of soft robots is interesting at several levels. First as a tool for design, to optimize prior the fabrication and reduce iterations of prototypes. Second, as a tool for solving the complex problem of motion control. FEM provides this accurate representation and has many advantages over most other numerical analysis methods, including versatility: there is no geometric restriction (in practice a very complex geometry can be difficult to mesh), FE mesh allows to improve the approximation (e.g. by adding points where large deformations occur and more resolution is required), material properties can differ from one element to the other, and others (Cook et al. 2007). The reader can also refer to (Nealen et al. 2006) for a nice survey on numerical analysis methods, for the computer graphics community.

#### Limitations

The principal drawbacks of FEM are the computational expenses. FEM simulations especially become costly when non-linear systems are involved. These non-linearities come from several levels: the non-linear deformations of the structure (rotation), and the non-linearity of the material law (non-linear elasticity). These computational expenses have often been a bottleneck for applications requiring real-time performance, as it is in the robotics community. Yet, computational power continue to increase and methods have been studied for improving the convergence speed and the stability of numerical solutions. For instance, work from computer graphics and medical simulation are showing FEM simulations with real-time performance. To mentioned just a few, work on the co-rotational model (Hauth & Strasser 2004), which deals with the geometrical non-linearities (non-linear deformations), and gives good approximations with the assumption of small deformations. The co-rotational model is a good solution to model linear elasticity. For non-linear elasticity and geometry requiring high resolution meshes, work on model reduction methods (Sifakis & Barbic 2012) can dramatically reduce the computation time. These methods consist in projecting the large FE system of equations onto subspaces of smaller dimensions. Note that a work on model order reduction for the control of soft robots has recently been proposed by our group (Goury & Duriez 2018).

For this thesis we choose to work with a numerical model of the soft robots using FEM. Indeed, we think FEM is a great solution to model a broad-spectrum of soft robots with good accuracy, in particular when no other simpler methods are available. Yet, the method is complex to implement. Some simulation softwares make it more easy by opening their code and allowing researchers to invest new methods without having to deal with the FEM implementation. We discuss these softwares in the following section.

### 1.3.2 Simulation software

In the field of rigid robotics, one can use either dedicated products like WorkspaceLt, RoboticSimulation, NI-Robotics, RoboNaut, SimRobot, or general purpose open-source software like Gazebo (Koenig & Howard 2004). The cited tools rely on off-the-shelves simulation kernels such as Open Dynamics Engine, Bullet Physics, NVidia PhysX or DART. These simulation kernels come from the video-game industry and are often focused on articulated-rigid bodies.

The same simulation kernels have been successfully used to model and simulate soft robots as in the NASA Tensegrity Robotics Toolkit (Caluwaerts et al.

2014) or in (Rieffel et al. 2009) with the use of PhysX to evaluate the candidate solutions of genetic algorithms. The video-game based simulation frameworks are fast and efficient to compute rigid-body simulations as well as some kind of soft bodies. They are also relatively easy to use, as required background knowledge in physical modeling is reduced. The counterpart is that very few of them are capable of modeling physically realistic deformable materials.

#### Realistic simulation

When a realistic deformable material simulation is needed, tools from the structural and multi-physics analysis field, as Abaqus or ComSol, are suitable. They rely on precise modeling formulations of continuous mechanics, and some of them are capable of handling multi-physics. The cost for such capabilities is the slow computation speed and the fact that a good understanding of physical modeling is required. The consequence is that they are only usable for simulation of soft robots, in combination with CAD software while designing the soft robotic parts. Simpler alternatives exist such as Voxelyze. Presented in (Hiller & Lipson 2014), it simulates soft materials undergoing large deformations and is associated with VoxCAD, a GUI simplifying the editing of the robot. Voxelyze relies on voxels to represent the object. It is used in (Cheney et al. 2014) to evaluate through simulation the walking capabilities of soft robots produced by genetic algorithms. In (Cheney et al. 2015), the same authors added interaction between the robot and its environment. Nevertheless, with a voxel simulation, it is not possible to approximate some geometrical shapes without an exaggerated number of voxels which leads to an increased computation time. In addition, with Voxelyze, the mechanical model is using beam theory on the lattice supporting the voxels. Such an approach may not capture the continuous material deformation in a realistic manner.

### 1.3.3 SOFA framework

For this work, we chose to use SOFA, an open-source framework for physics-based simulation. Its development has first been motivated by the field of surgical and biomedical simulation (Faure et al. 2012). The interesting point of this framework is the focus on deformable objects and complex interactions. A tool like SOFA can simulate a large choice of mechanical models: from rigid-bodies or mass-spring, to one implementing realistic hyper-elastic material with FEM. It can operate on a wide range of geometrical descriptions from 1D (curve) and 2D (surface mesh) to 3D (voxels, multi-resolution octrees or hexahedral and tetrahedral mesh), see Figure 1.12. It is capable of handling collisions and contacts precisely, as well as to handle multi-physics behaviors

(Talbot et al. 2012, 2015). It is also capable of interacting with sensing hardware (Kinect, OptiTrack, LeapMotion), that are commonly used in robotics, as well as with haptic devices (Peterlik et al. 2011). A framework like SOFA can be considered as a *bridge* between video-game and structural analysis approaches.

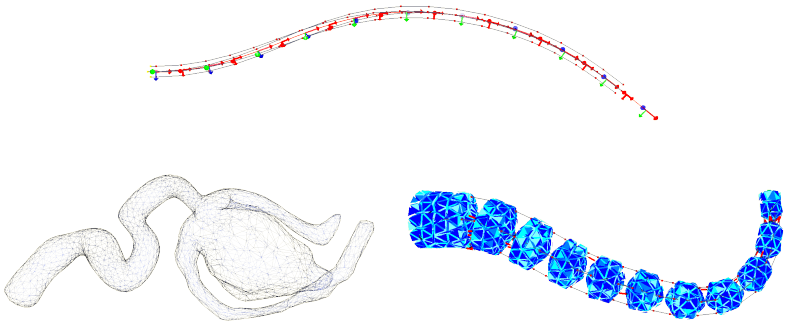

Figure 1.12: *SOFA simulations. (top) 1D beam representation of a soft catheter actuated with cables. (bottom-left) 2D shell representation of a pressurized vessel with an aneurysm. (bottom-right) 3D tetrahedral representation of our soft trunk actuated with cables.*

The framework is mostly intended for research purposes. It is quite easy to connect new methods to the existing implementations of SOFA. In a way similar to Gazebo (Koenig & Howard 2004), SOFA has a scene-graph based simulation architecture, which is commonly used in 3D modeling tools. A scene contains the robot and its environment and is described in XML or with a Python script. SOFA is also a component based architecture. A robot is then an assembly of elementary components: some components are for rendering, others for contact or topology encoding, others for numerical integration or mechanical modeling.

SOFA also introduced multiple model representation. Multi-model means that in a simulated object a property can be represented in multiple ways. A good example is the shape of a robot: a low resolution tetrahedral mesh can be used for FEM, while using a high resolution triangular mesh for the rendering, and a low resolution triangle mesh for the contact management. To connect the representations, SOFA introduces *mappings*. Given the position of the DoFs $q$ of a representation, one can define a second representation with $x = M(q)$, where $M$ is a possibly non-linear mapping function. The strength of *mappings* is that constraints on a representation can be transferred to a second representation. Using this tool, we can gather actuators, effectors and contacts from different representations to obtain a full system, without having to change the implementation of the components. They are defined regardless of the geometrical dimension of the object (1D, 2D, 3D), geometrical

representation (voxel, curve, octree, tetrahedral mesh) or mechanical model. As already mentioned in section 1.2.1, if the robot is hybrid, SOFA offers a way to combine deformable model and rigid ones. This is done through this *mapping* mechanism which allows to transfer positions, velocities and forces between the deformable model and the rigid components. An other strength of SOFA is that it provides real-time performance for simulations of around 6000 DoFs (corresponding to 2000 nodes for a deformable object), which is sufficient for robots with simple geometry. Also, a GPU implementation is available for more complex geometry.

#### SoftRobots plugin

For this thesis we developed a plugin for SOFA dedicated to soft robots. It contains useful tools for the modeling, simulation and motion control of soft robots. It is presented in (Coevoet, Morales-Bieze & et al. 2017). The plugin is divided into two parts: (1) *SoftRobots plugin*: This part is open and hosted on github. It contains a lot of tools allowing to simulate the forward kinematics and dynamics of soft robots, with a large possibility of actuations. (2) *SoftRobots.Inverse plugin*: This second part is private. It contains all the tools necessary to solve the IP of soft robots motion control, and is an addition to the first part. All type of actuation proposed in the first plugin can, in the latter one, be optimized to solve the IK and ID problems. All the methods proposed in this manuscript are available in the plugin.

## 1.4 Contributions

As mentioned in section 1.2.4, there is a need of new techniques for the modeling and motion control of soft robots. Moreover these controllers should handle contacts, as most of robotics applications involve interactions with an environment. The contributions of the thesis are listed below:

- **Inverse model of deformable structures**: we first propose a solution, based on FEM for the problem of controlling the motion of soft structures, and in particular the motion of soft robots. We formulate the problem as an optimization problem with a quadratic form, a Quadratic Program (QP). We aim at providing generic solutions, that is without particular geometrical nor material assumptions. We also focus on proposing controllers with real-time performance. The results have been published in (Coevoet, Morales-Bieze & et al. 2017), and are presented in chapter 3.
- **Inverse model with contact handling**: the formulation of the optimization when dealing with contacts takes the form a Quadratic Program

with (linear) Complementarity Constraint (QPCC), as we use Signorini's law to model the contact. We propose a solver to solve this problem in real-time and allow to control the motion of soft robots evolving in a moving and frictionless environment. The results have been published in (Coevoet, Escande & Duriez 2017), and are presented in chapter 4.

- **Inverse model with stick contact handling**: the addition of friction introduces non-linear complementarity conditions which makes the problem more complex. With the assumption of sticking contact only, we propose a real-time solution that opens the control of some soft robots locomotion and grasping. The results have been published in (Coevoet et al. 2018), and are presented in chapter 5.

The usefulness of this work is demonstrated with direct applications for the medical field and for the entertainments and the art. The list of publications is given at the end of the manuscript.

## 1.5 Organization of the manuscript

The remainder of the manuscript is organized as follow. The second chapter is dedicated to the state of the art, regarding the contributions of the thesis. We first discuss the current solutions proposed to solve the IK and ID problems of actuated structures in general. Second, we discuss in particular work that handle contacts in their controller. In the third chapter we present in detail the algorithms we propose for soft robots IK and ID solutions, including detail about the modeling, the optimization problem, and some direct applications of the method. The fourth chapter is given to contact handling, where we detail formulations and the solver we propose to solve this complex problem with real-time performance. We also present some direct applications. The fifth chapter is dedicated to the addition of friction into the problem, and in particular the control of soft robots locomotion and grasping. In each of these three latter chapters, we provide numerical and real experiments to show the efficiency of the proposed methods. We finally conclude the thesis in the sixth chapter.

Chapter 2

# State of the Art: Inverse Model and Contact Handling

## Contents

## 2.1 Introduction

Controlling the motion of a continuum or a soft robot is challenging, as it involves to deal with continuous deformations of the robot structure. In this chapter we review the contribution on IK and ID for the motion control of actuated bodies (hard or soft). These problems have been investigated by the robotics and the computer graphics communities. The reader can refer for example to, a book on control for traditional rigid robots (Craig 2005), a recent survey on control strategies for continuum and soft robotics (Thuruthel et al. 2018), and another great recent survey on IK problems from the computer graphics community (Aristidou et al. 2017).

In computer graphics, IK and ID are used to make animation of virtual actuated characters physically plausible, and then enhance the esthetic's of animations. These characters are either articulated figures, soft bodies with embedded skeleton, or entirely soft bodies actuated with virtual muscles. They thus share the problematics of both traditional and soft robotics communities. Though they generally do not have to deal with real prototypes, nor real actuators, the algorithms they propose are of interest for us.

In each section of this chapter, we discuss the solutions proposed by both robotics and computer graphics communities. The state of the art is reviewed as follow: First, we look at general IK and ID solutions for actuated structures in section 2.2. Second, we review IK and ID solutions with contact handling in section 2.3.

## 2.2 Inverse kinematics and inverse dynamics

Different methods are used to solve IK and ID problems of actuated structures. In this section we review analytical, Jacobian-based, optimization-based and data-driven learning methods. Because of the vastness of the subject, this is not an exhaustive review. For the sake of clarity, all presented publications are gathered and listed in Tables 2.1, 2.4, 2.2, 2.3, and 2.5 with respect to the IP resolution method proposed, the modeling method used, and the system controlled.

### 2.2.1 Analytical solutions

Traditional robot manipulators are often designed so that analytical (or closed-form) solutions to the IK problem exist (see (Craig 2005) or (Paul 1981) for more detail). Also in computer animation, where several researchers have addressed the IK of anthropomorphic arm and leg with analytical approach.

For example, Tolani & Badler (1996) provides an analytic solution for a 7DoFs simple model of the human arm, and a combination of analytical and numerical methods for an anthropomorphic arm in (Tolani et al. 2000). For hyper-redundant robots, and in particular soft robots, it is more difficult and often impossible to get such solutions without making strong assumption on the robot geometry. However, analytical solutions are appreciated for their low computational cost, and because they are reliable and offer a global solution.

### Constant curvature models

For particular continuum robots, that is robot with rod-like shapes, constant curvature model (for single-section) and piecewise constant curvature model (for multi-section) have been proposed and widely employed to get analytical IK solutions (Hannan & Walker 2003) (Jones & Walker 2006) (Duindam et al. 2010). A review can be found in (Webster & Jones 2010).

The models are based on a geometric approach that significantly simplifies the problem. The idea is to assume that each section of a multi-section rod-like robot or arm, deforms with constant curvature. For example, in (Sears & Dupont 2006), Sears & Dupont present an analytic solution to the IK problem of steerable concentric tubes, based on a piecewise constant curvature representation of the tubes. In (Neppalli et al. 2009), Neppalli et al. propose an analytic solution to the IK problem of a multi-section continuum trunk, based on a constant curvature description of each section of the robot. They make additional simplifications by assuming that gravity loading produces no deformation on the robot, which often leads to quite inaccurate solutions. This is usually the case with analytical solutions.

While these methods are simple and fast, not all continuum robots suit the assumption of segmented constant curvature.

### Conclusion on analytical solutions

As mentioned, for most soft robots, it is hard to get analytical solutions that describe the mechanical behavior with accuracy. Expect for continuum robots with particular geometry (i.e. rod-like shape), that can be approximate with constant curvature models. However, these methods have difficulty coping with environmental changes. The alternative is to use numerical methods, and iteratively approximate a good solution to the problem. In the following, all the methods we present fall into the class of numerical analysis.

| Analytical solutions | | | |
|---|---|---|---|
| Publication | Method | Model | System |
| (Tolani & Badler 1996) | Analytical (IK) | Articulated rigid model | Rigid manipulators |
| (Tolani et al. 2000) | Combination of analytical & numerical methods (IK) | Articulated rigid model | Rigid manipulators |
| (Hannan & Walker 2003) | Analytical (IK) | Piecewise Constant Curvature | Continuum trunk manipulator |
| (Jones & Walker 2006) | Analytical (IK) | Piecewise Constant Curvature | Multisection Continuum Robots |
| (Duindam et al. 2010) | Analytical (IK) | Piecewise Constant Curvature | Steerable continuum Needles |
| (Sears & Dupont 2006) | Analytical (IK) | Piecewise Constant Curvature | Steerable continuum concentric tubes |
| (Neppalli et al. 2009) | Analytical (IK) | Piecewise Constant Curvature | Multi-section continuum trunk |

Table 2.1: *Publications are categorized according to the method used to solve the IP, the model used and the controlled system. Results for continuum or soft robotic systems are softly highlighted in gray.*

### 2.2.2 Jacobian-based methods

Jacobian-based methods are the first numerical methods that we present. The Jacobian matrix gives the differential relationship between actuators variables and end-effectors position. By inversing this matrix, one can solve the IK problem. However, the Jacobian matrix can be neither square (the number of actuators variables is larger that the number of end-effectors positions, or vice versa) nor invertible (when singularities occur). IK singularities occur when there exist multiple solutions, that is multiple actuators configurations for a single end-effector position. There exist several methods based on the Jacobian matrix, such as Jacobian pseudo-inverse methods, Jacobian transpose methods, and damped least squares methods. In the following, we discuss some of these methods. A great review on Jacobian methods for computer graphics and real-time animation is given by Buss in (Buss 2004).

#### Jacobian pseudo-inverse

A generalized inverse of the Jacobian matrix is used when its inverse does not exist, that is when the matrix is singular or rectangular. In (Whitney 1969), Whitney investigate the use of the Jacobian pseudo-inverse to solve the IK of rigid manipulators. In (Zhang et al. 2016, 2017), Zhang et al. propose

closed-loop systems for the control of soft robots. A Jacobian matrix for soft robots is formulated, based on the same FE modeling than we use in this thesis, and a pseudo-inverse of the Jacobian matrix is then used to solve their IK. Pseudo-inverse methods tend to have stability problems in the neighborhoods of singularities (Buss 2004): when the configuration is close to a singularity, large changes in actuators variables may occur, even for small movements in the target position. Other methods can be used to overcome this stability problems, such as damped least squares methods.

### Damped least squares

Damped least squares methods are local optimization methods that can prevent infeasible solutions near singular configurations (Deo & Walker 1995). A review on these methods for robot manipulators IK can be found in (Deo & Walker 1995). In (Nakamura & Hanafusa 1986), Nakamura & Hanafusa introduce the Singularity Robust inverse method as an alternative to the pseudo-inverse of the Jacobian matrix. The method gives solution even when inverse and pseudo-inverse matrices are not feasible at, or in the neighborhood of singular points. They apply the controller to rigid manipulators. In (Baerlocher & Boulic 1998), Baerlocher & Boulic deal with task prioritization for the kinematic control of human-like articulated figures (virtual figures). They use damped least squares methods to solve the IK of articulated figures for the purpose of posture control and design. An extension of damped least squares called selectively damped least squares for IK solutions of articulated bodies is proposed in (Buss & Kim 2005). In (Harish et al. 2016), Harish et al. propose a parallelized design of the damped least squares IK for articulated bodies, and with prioritized end-effectors. The authors state that the algorithm is highly scalable and can handle complex articulated bodies at interactive frame rates. Note that for these methods to be stable, a constant damping factor has to be chosen carefully.

### Other Jacobian methods

Other methods such as Jacobian transpose methods are interesting against pseudo-inverse methods in the sens that, no matrix inversion is required, and thus an iteration can be performed very quickly. However, these methods have poor convergence properties (Baerlocher 2001). Schreiber & Hirzinger use a Jacobian transpose method to solve the IK problem of a rigid manipulator. They introduce an heuristic measure for treating kinematic singularities in (Schreiber & Hirzinger 1998). A Jacobian method is compared against other methods in several works, for example (Aristidou et al. 2017). In (Giorelli et al.

2012), Giorelli et al. propose a method based on the Jacobian inverse and solve the IK problem numerically. They built a soft manipulator with a conical shape, inspired by the octopus arm. The IK do not take into account the effect of gravity on the manipulator. The robot is made of silicone and actuated with cables underwater. The authors stipulates that the gravity force and buoyancy, balance each other due to the similarity between the density of the silicone and that of water. The results are almost accurate, and the IK solver shows quite slow performance, as stipulate in (Giorelli et al. 2015). In (Tevatia & Schaal 2000), Tevatia & Schaal propose an extended Jacobian method for the real-time control of highly redundant robots, like humanoid robots.

| Jacobian-based methods | | | |
|---|---|---|---|
| Publication | Method | Model | System |
| (Whitney 1969) | Jacobian pseudo-inverse (IK) | Articulated rigid model | Rigid manipulators |
| (Zhang et al. 2016, 2017) | Jacobian pseudo-inverse (IK) | FEM | Soft robots |
| (Nakamura & Hanafusa 1986) | Singularity Robust inverse Jacobian (IK) | Articulated rigid model | Rigid manipulators |
| (Baerlocher & Boulic 1998) | Damped least squares (IK) | Articulated rigid model | Virtual articulated characters |
| (Buss & Kim 2005) | Damped least square (IK) | Articulated rigid model | Virtual articulated characters |
| (Harish et al. 2016) | Damped least square (IK) | Articulated rigid model | Virtual articulated characters |
| (Schreiber & Hirzinger 1998) | Jacobian transpose (IK) | Articulated rigid model | Rigid manipulators |
| (Giorelli et al. 2012) | Jacobian inverse (IK) | Cosserat rod | Soft manipulator actuated underwater |
| (Tevatia & Schaal 2000) | Extended Jacobian (IK) | Articulated rigid model | Humanoid robots |

Table 2.2: *Publications are categorized according to the method used to solve the IP, the model used and the controlled system. Results for continuum or soft robotic systems are softly highlighted in gray.*

### Conclusion on Jacobian methods

Jacobian methods are popular methods in the robotics and computer graphics communities. Each method have advantages and drawbacks over the others. In general, these methods are liked for their simplicity and their low computational expend. However, a lot of these methods suffer from stability problem due to singularities, and incorporating constraints into these methods is not

straightforward. Some effort has been made to deal with actuation limitations (e.g. limits on cable displacement), see (Aristidou et al. 2017) for references. Other methods, such as optimization-based methods, are better suited to deal with singularities and constraints on actuators. In the next section we discuss those solutions.

### 2.2.3 Optimization-based methods

The IK and ID problem can be formulated as a constrained optimization (minimization or maximization) problem, also called programming problem. Models with large DoFs can be considered, and more design variables can be included in the optimization formulation. Singularity problems are often tackled by introducing additional constraint into the program or additional terms in the objective function. A large number of resolution methods have been developed to solve such constrained problems, together with products and open libraries that can be used as third-part. For quadratic program, we have used the open-source C++ library qpOASES. Depending on the nature of the program, typically linear or non-linear (objective function and/or constraints), the resolution may be very expensive. Note that in most cases, searching for a global solution is too expensive, so for real-time applications, local optimization methods are generally preferred. It is then possible to be stuck in a local optimum of the objective function. However, optimization-based methods usually return smooth motion without erratic discontinuities (Aristidou et al. 2017).

#### Non-linear programming

A non-linear program denotes the large class of programs which have at least, either the objective or some constraints being non-linear functions. These problems are usually more difficult to solve in real-time than linear programs. In (Zhao & Badler 1994), Zhao & Badler solve the IK of highly articulated figures by formulating a non-linear programming subject to linear constraints corresponding to the joints limits. Sifakis et al. build an anatomically accurate model of facial musculature, passive tissue and underlying skeletal structure in (Sifakis et al. 2005). They propose to determine muscle activations using a non-linear programming. In (Skouras et al. 2013), Skouras et al. optimize, among others, the location of cables on deformable objects to reach target animations. They model the objects with FEM and they use an augmented constrained optimization to formulate the IP. The final actuation (cables displacement) seems to be directly derived from the animation, that is by computing the cables length knowing their location on the animation. Note

that this technique cannot work with cables passing through the structure. Similarly Bern et al. propose in (Bern et al. 2017) a method, reminiscent of IK, for the creation of animated soft tendon-driven toys: the location of the tendons path are computed so that the toy can perform desired motions.

### Quadratic programming

A QP is a particular type of non-linear programing. It corresponds to an optimization program whose objective function has a quadratic form, and whose constraints are linear. In (Sueda et al. 2008), Sueda et al. propose a method to generate the motion of tendons and muscles under the skin of a traditionally animated character, and thus enhance the realism of hand simulations. The skeleton and tendons are simulated using a musculoskeletal model. The skin is attached to the skeleton, and the subcutaneous deformation from tendon motion is added as a post-process. They formulate the IP of tendons activation as a quadratic optimization. In (Coevoet et al. 2014, 2015), we propose to register anatomical deformable structures for the context of adaptive radiotherapy, and using a quadratic optimization. The method is based on a FEM modeling of the soft structures. The problematic and results of this work are discussed in detail in section 3.5.1. The same quadratic optimization is used in (Largilliere et al. 2015) and (Rodríguez et al. 2017) for the control of soft robots, where in the first paper we propose a multi-rate solving of the IP, and in the second we propose to solve the IK of soft robots actuated with hydraulic. In (Coevoet, Morales-Bieze & et al. 2017), we propose a unified framework for the motion control of soft robots, again based on FEM and quadratic programming. The algorithms are detailed in chapter 3.

### Other optimization-based methods

Cyclic coordinate descent method is an other popular optimization-based method to solve the problem of articulated chains IK. The approach consists on an iterative resolution of optimization problems, that is by solving the transformation of one joint variable at a time in a cyclic manner. See (Wang & Chen 1991) for the first introduction of the method, where Wang & Chen apply it on the control of a rigid manipulator.

In (Duriez 2013), Duriez proposes to solve the motion control of soft robots with a method based FEM, and an iterative Gauss-Seidel algorithm to solve the IK. The algorithm iterates over the actuators variables, optimizing each variable at a time while freezing the others.

| Optimization-based methods | | | |
|---|---|---|---|
| Publication | Method | Model | System |
| (Zhao & Badler 1994) | Non-linear program (IK) | Musculoskeletal: Articulated rigid model, FEM | Virtual facial musculoskeletal |
| (Sifakis et al. 2005) | Non-linear program (IK) | Articulated rigid model | Virtual articulated characters |
| (Sueda et al. 2008) | QP (IK) | Musculoskeletal: Articulated rigid model, spline-based strands | Virtual tendon activation |
| (Coevoet et al. 2014, 2015) | QP (IK) | FEM | Soft structures |
| (Largilliere et al. 2015) | QP (IK) | FEM | Soft robots |
| (Rodríguez et al. 2017) | QP (IK) | FEM | Soft robots (hydraulic) |
| (Coevoet, Morales-Bieze & et al. 2017) | QP (ID) | FEM | Soft robots |
| (Wang & Chen 1991) | Cyclic coordinate descent (IK) | Articulated rigid model | Rigid manipulators |
| (Duriez 2013) | Gauss Seidel (IK) | FEM | Soft robots |

Table 2.3: *Publications are categorized according to the method used to solve the IP, the model used and the controlled system. Results for continuum or soft robotic systems are softly highlighted in gray.*

#### Conclusion on optimization-based methods

Optimization-based methods are particularly suited for constrained problems. Indeed we will see in the following section 2.3, where we review solutions for the special case of contact handling (often formulated as a constrained problem), that optimization-based methods are more employed than the others. However, although these methods have great stability and convergence properties, they can be computationally expensive depending on the complexity of the problem. Still, it is the direction we took in this thesis.

### 2.2.4 Data-driven learning methods

Data-driven learning methods provide another approach that can be less expensive to solve IK problems. The idea behind these methods is to constitute a database from pre-learned configurations, or pre-learned postures, and use it to find a feasible pose that match end-effectors target. The methods involve two

computation steps, first at training time to constitute the database (typically offline), and second at runtime to provide a solution. Some methods also propose to learn at runtime, and thus continuously adapt the control to a changing environment. This is essential for soft robots since, as already mentioned, the environment has a big influence on their behaviors. We discuss these particular works in section 2.3. Data-driven methods are appreciated because they are often easy to implement, and fast at runtime (allowing real-time control). Note that singularities are handle during the learning phase, using regularization methods. See (Omidvar & Van der Smagt 1997, Nguyen-Tuong & Peters 2011) for reviews on IK and/or ID solutions using learning methods for traditional robotics control, and see (Aristidou et al. 2017) for computer graphics applications.

#### Feed-forward neural network

Feed-forward neural network has been used in recent work to find IK solutions of continuum and soft robots. Note that *Feed-forward* simply refers to the fact that there is no cycles or loops in the *network*. The idea of these methods is to directly learn the IK mapping between end-effectors location and actuator variables. We think the earliest work using learning methods for solving ID of continuum robots was proposed by Braganza et al.. In (Braganza et al. 2007), Braganza et al. propose a controller based on a feed-forward neural network which can deal with the uncertainty in the structure of continuum robot's dynamic model.

In (Giorelli et al. 2015), Giorelli et al. compare two approaches for solving the IK of a cable-driven soft arm with non-constant curvature. The first approach is the Jacobian-based method from (Giorelli et al. 2012), while the other is the feed-forward neural network from (Giorelli et al. 2013). The performance of the two methods are compared in term of accuracy and computational time. The learning method appears to be more effective and gives real-time performance at runtime. In (Thuruthel et al. 2016), Thuruthel et al. propose to learn the global IK solutions of a soft manipulator, using a feed-forward neural network (a multilayer perceptron with single hidden layer). In (Jiang et al. 2017), Jiang et al. propose a two levels method to the solve the IK of a soft arm. This two levels method involves an analytical solution to solve the mapping from task space to configuration space, and a neural network to solve the mapping from configuration space to actuator space. Or see also (Melingui et al. 2014, 2015).

### Gaussian processes regression

Some other learning methods that fall into the class of statistical methods are also proposed. For example, in computer graphics, the problematic of generating plausible and natural skeleton animations, has raised IK solutions using Gaussian processes regression. The goal is to allow animators to generate natural poses by giving target location of only few points on the character. In (Grochow et al. 2004), Grochow et al. propose a real-time IK that learns model of human poses, based on the called scaled Gaussian process latent variable model. Given some examples motion data, the method learns an objective function from all the poses, and then an optimization is ran with the new constraints to generate natural poses. Similarly, in (Holden et al. 2015), Holden et al. use a Gaussian process regression and also provide solutions with real-time performance. Note that as stated in (Grochow et al. 2004, Holden et al. 2015), these methods are limited to small problems and small data sets. Gaussian process regression method has also been applied for the control of traditional rigid robots (see examples in (Nguyen-Tuong & Peters 2011)).

### Deep neural network

Deep neural network has been used to solve more complex motion and high level task problems. Note that *Deep* refers to the fact that the neural network has multiple hidden *layers*, and thus has a higher lever of abstraction. In (Holden et al. 2016, 2017), Holden et al. develop a variety of deep neural networks to synthesize and edit animation of complex human motions. In (Polydoros et al. 2015), Polydoros et al. propose to learn the ID of a robotic manipulator by employing a real-time deep learning algorithm. The authors stipulate that the proposed algorithm should be applicable for online learning. In (Phaniteja et al. 2018), Phaniteja et al. propose to learn the IK of humanoid robots and generate dynamically stable solutions, using deep reinforcement learning.

### Conclusion on data-driven learning methods

The main difficulty with data-driven learning methods is to generate a database sufficiently rich, i.e. a database that contains as much relevant informations about the system as possible (Nguyen-Tuong & Peters 2011). Not to mention that most learning approaches utilize a supervised learning technique for training, that is a labeling of training data, which can be expensive to get (Nguyen-Tuong & Peters 2011). Together with this long phase for data collection, the main inconvenient with these methods is that it is hard to predict or even analyze the behavior of the network after learning (although

preliminary tests can be conducted on simulated models). Also, changing a single value or parameter of the robot (e.g. in the design, weight, stiffness) will require to redo the learning phase entirely.

However, despite these few drawbacks, data-driven methods are appreciated because they are often easy to implement, and very fast at runtime (allowing real-time control).

| Data-driven learning methods | | | |
|---|---|---|---|
| Publication | Method | Model | System |
| (Braganza et al. 2007) | Feed-forward neural network (ID) | - | Continuum manipulator |
| (Giorelli et al. 2013, 2015) | Feed-forward neural network (IK) | Cosserat rod (with Euler-Bernoulli hypothesis) | Cable-driven soft arm |
| (Thuruthel et al. 2016) | Feed-forward neural network (IK) | Cosserat rod (with Euler-Bernoulli hypothesis) | Soft manipulator |
| (Jiang et al. 2017) | Analytical & Feed-forward neural network (IK) | Piecewise constant curvature | Soft arm |
| (Melingui et al. 2014, 2015) | Adaptive neural network (IK) | - | Continuum manipulator |
| (Grochow et al. 2004) | Scaled Gaussian process latent variable (IK) | Articulated rigid model | Virtual articulated characters |
| (Holden et al. 2015) | Gaussian process regression (IK) | Articulated rigid model | Virtual articulated characters |
| (Holden et al. 2016, 2017) | Deep learning (IK) | Articulated rigid model | Virtual articulated characters |
| (Polydoros et al. 2015) | Deep learning (ID) | Articulated rigid model | Rigid manipulator |
| (Phaniteja et al. 2018) | Deep reinforcement learning (IK) | Articulated rigid model | Humanoid robots |

Table 2.4: *Publications are categorized according to the method used to solve the IP, the model used and the controlled system. Results for continuum or soft robotic systems are softly highlighted in gray.*

### 2.2.5 Other methods

In (Aristidou & Lasenby 2011), Aristidou & Lasenby propose a method, called forward and backward reaching IK for the control of articulated figures. The

method involves iterations between successive forward and backward modes, during which the joints point are moved independently and one by one along the chain. The method handles joint limitations. Note that they compare the method with the most popular Jacobian-based methods regarding reliability, computational cost and conversion criteria. The proposed method shows better and faster results from their examples.

In (Bryson & Rucker 2014), Bryson & Rucker propose a parallel continuum manipulator, which design is similar to a rigid-link Stewart-Gough platform. They model each leg of the manipulator using Cosserat rod model, and solve the IK with a shooting method. The method they propose deals with external loads. In (Till et al. 2015), Till et al. detail methods employ for real-time performance of the latter algorithm.

| Other methods | | | |
|---|---|---|---|
| Publication | Method | Model | System |
| (Aristidou & Lasenby 2011) | Forward and backward reaching method (IK) | Articulated rigid model | Virtual articulated characters |
| (Bryson & Rucker 2014) | Shooting method (IK) | Cosserat rod | Parallel continuum manipulator |

Table 2.5: *Publications are categorized according to the method used to solve the IP, the model used and the controlled system. Results for continuum or soft robotic systems are softly highlighted in gray.*

### 2.2.6 Conclusion on IK and ID methods

As we have seen in the previous sections, there are numerous methods to solve IK and ID problems, especially for traditional robotics and virtual figures. Each of them has its own advantages and drawbacks, and a close look to these differences should be taken before choosing which one to use. They can be compared on the following criteria: efficiency (computational speed), accuracy, robustness, generality (e.g. robot's shape, actuations), and complexity of the method (implementation).

The choice of a method clearly depends on the possibilities that the robotic system offers, and the application. However, we choose to work with a numerical model of the soft robots using FEM to provide generic solution, and to formulate the problem as an optimization program to allow for complex control. Indeed, as a reminder, optimization provides smooth solutions and are well adapted to deal with constraints on actuation and singularity problems.

## 2.3 IK and ID with contact handling

As widely mentioned in chapter 1, most soft robotic applications involve the robot to interact with its environment. Then, it becomes essential for soft robots controller to account for contacts. This makes the problem much more complicated, in particular when dealing with friction.

Friction effects are the central key for locomotion and grasping tasks. Numerous works from the computer graphics community looks at ID problem for the locomotion of animated (hard and soft) virtual characters. For traditional robotics, this problematic has also been largely studied for locomotion and manipulation tasks.

In the following, we discuss the methods we have found in the literature that handle collision within the IP. Again, for the sake of clarity, all the presented publications are gathered and listed in Table 2.6 with respect to the IP resolution method proposed, the modeling method used, and the system they control.

### 2.3.1 Jacobian-based methods

A model-less approach is proposed in (Yip & Camarillo 2014) to control the end-effector position of continuum robots interacting into an environment with unknown obstacles. The method relies on an empirical estimate of the robot Jacobian computed first offline, and then updated online using measurements on actuators and the end-effector position, together with an optimization resolution. They test their controller on a cable-driven continuum robot. While the method offers a closed-loop task-space control that is very simple to implement, it is designed for static environment only. In (Yip & Camarillo 2016), Yip & Camarillo extend the previous work to control both end-effector position and forces.

### 2.3.2 Optimization-based methods

We start this section with the contributions of the computer graphics community. In (Kim & Pollard 2011*a*), Kim & Pollard propose to control simulated articulated deformable characters, by combining an active skeleton simulation with a passive FE simulation of the deformable structure. They use a penalty-based contact model to simulate contact normal forces. Penalty-based contact model have several drawbacks. The idea is to create a spring between the two object at the contact point (usually an implicit spring model is used for stability purposes), which gives a coupled system of the two colliding objects. Thus, this method involves to solve large systems of equations. Furthermore,

this technique is not physically correct, and the results highly depend on the stiffness chosen for the springs. It may result to interpenetration of the objects (if the stiffness is too low), or hard take off (if the stiffness is too high). For the friction effect, Kim & Pollard use Coulomb's law, which is physically accurate, but it is not clear for us how this constraint is solved within the inverse algorithm. In (Kim & Pollard 2011*b*), Kim & Pollard detail strategies employed for real-time or near real-time performance, including a mesh embedding technique and a parallel computation. In (Liu et al. 2013), Liu et al. propose a quite similar approach, but extend the work of Kim & Pollard by achieving the control of human like characters with soft body. To do so, they extend the motion control strategy from (Liu et al. 2010) by using a trajectory-based ID and a time scaling scheme to augment the optimization.

In (Tan et al. 2012), Tan *et al.* use FEM and muscle fibers actuation to simulate a soft body character's locomotion. The ID problem of the soft characters is formulated as a QPCC. They use Coulomb's law and a constraint based model for contacts. They are able to handle stick and sliding effects, but they have low computation performance due to the expensive nature of their solver. In (Coros et al. 2012), Coros et al. dynamically optimize the rest shape of soft bodies to induce desired deformations. They use a penalty-based contact model for normal and tangential forces, and formulate the IP as a non-linear program that they solve using a sequential QP. Their characters are able to crawl, roll, propel themselves and walk. In addition, they propose an example-based strategy that can be used to allow for artistic control over the resulting motions.

From the robotics community, optimization-based methods have been widely used to control locomotion and manipulation tasks of humanoid rigid robots. For example, in (Kim et al. 2008), Kim et al. formulate the locomotion problem of humanoid robots as a non-linear program, that they solve using a sequential QP. In (Herzog et al. 2016), Herzog et al. solve hierarchical ID based on cascades of QPs for the control of legged robots. Note that the complementarity condition for contact, to switch between active and inactive contact, is often not formulated. Instead they use different techniques to estimate the contact points to set as hard constraints. When in contact, they both use Coulomb's law for friction. Other work, like Posa et al., use a complementarity formulation to enable control of robots that are frequently making and breaking contact, in an unpredictable manner. They formulate the problem as a mathematical program with complementarity constraints, and use a sequential QP to solve the problem, leaving real-time performance as future work.

For complex multi-tasks, Escande et al. use specific hierarchical quadratic programming that enables fast computations (Escande et al. 2010, 2014).

Hierarchical optimization approaches are often used in robotics when multiple and sometimes incompatible objectives are involved. For example, with a humanoid robot, they can seek to solve the problems of walking, collision avoidance and manipulation at same time. The particularity of the algorithm proposed in (Escande et al. 2014) is that it solves the complete hierarchical problem at once, and not a cascade of QP as it may often be the case.

In (Coevoet, Escande & Duriez 2017), we propose to solve the IK problem of soft robots with Signorini's condition for contact. The problem is formulated as a QPCC and solved in real-time using a solver based on decomposition method. The algorithms are detailed in chapter 4. In (Coevoet et al. 2018), we extend the method to friction contact (stick case only), allowing the control of some soft robots rolling motions, and the control of object manipulation. Algorithms are detailed in chapter 5.

### 2.3.3 Data-driven learning methods

A learning approach for ID with contact handling has been proposed by Calandra et al. in (Calandra et al. 2015). The method they propose is based on a data-driven mixture-of-experts learning approach using Gaussian processes. The idea is to learn offline the ID when contacts occur, from experiments on the real robots. They evaluate their approach on the arm of a humanoid robot (*iCub*). Their approach is clearly limited, as unlearned contacts lead to bad results. In (Thuruthel et al. 2017), Thuruthel et al. introduce a machine learning based approach for closed-loop IK control of continuum manipulators. They use a feedforward neural network to learn the mapping between task space and actuators variables. Feedback information about the tracking error, due to unstructured environment, are used to correct the actuation without any modification of the learned network. The limitation here is that, they do not take the effect of contact into account somehow in the model. So if using the contact may lead to a better solution, or if the contact makes the error increase, it will not be captured.

### 2.3.4 Conclusion on contact handling

As we have seen, the problem of motion control of actuated systems, with contact handling, is of great interest for both robotics and computer graphics communities. For soft robots, it is a particular challenge that have been address only recently. On the other hand, contact problems, especially locomotion and manipulation control have been widely addressed for the animation of virtual characters, and for the control of rigid articulated systems (e.g. manipulators and humanoid robots). As we have seen, the most popular methods to solve

| Contact handling | | | |
|---|---|---|---|
| Publication | Method | Models | System |
| (Yip & Camarillo 2014, 2016) | Online Jacobian estimation (IK) | Model-less | Soft cable-driven robot |
| (Kim & Pollard 2011*a*,*b*) | QP (ID) | FEM, Articulated rigid model & penalty-based contact model | Virtual soft articulated characters |
| (Tan et al. 2012) | QPCC (ID) | FEM & constraint-based contact model | Virtual soft active characters |
| (Coros et al. 2012) | Non-linear program (ID) | FEM & penalty-based contact model | Virtual soft active characters |
| (Liu et al. 2013) | Trajectory-based (ID) | FEM, Articulated rigid model & penalty-based contact model | Virtual soft articulated characters |
| (Kim et al. 2008) | Non-linear program (ID) | Articulated rigid model & constraint -based contact model | Humanoid robots |
| (Herzog et al. 2016) | Hierarchical QP (ID) | Articulated rigid model & constraint -based contact model | Legged robots |
| (Posa et al. 2013) | Mathematical problem with complementarity constraint (ID) | Articulated rigid model & constraint -based contact model | Legged robots |
| (Escande et al. 2010, 2014) | Hierarchical QP (IK) | Articulated rigid model & - | Humanoid robots |
| (Coevoet, Escande & Duriez 2017) | QPCC (IK) | FEM & constraint -based contact model | Soft robots |
| (Coevoet et al. 2018) | QPCC (ID) | FEM & constraint -based contact model | Soft robots |
| (Calandra et al. 2015) | Mixture-of-experts learning (ID) | Model-less | Rigid arm |
| (Thuruthel et al. 2017) | Feedforward neural network (IK) | Model-less | Continuum manipulator |

Table 2.6: *Publications are categorized according to the method used to solve the IP, the model used and the controlled system. Results for continuum or soft robotic systems are softly highlighted in gray.*

such problems are based on quadratic programming. Indeed, the formulation of optimization programs is particularly suited for such highly constrained tasks. In this thesis we follow the same path by applying quite similar approach to the special case of soft robots control.

We propose two approaches to deal with contact. First, for applications (e.g. catheter guidance into blood vessels) where friction effects can be neglected. We propose to formulate the problem as a QPCC, as done in some previous work, and propose to use a specific solver with real-time performance, based on decomposition method. The details are given in chapter 4. Second, for the control of soft robots locomotion and manipulation tasks, where it is required to solve friction effects. The inclusion of friction makes the problem even more complicated. Indeed, if we want the results to follow physic's laws, we thus have to solve Coulomb's conditions, which lead to a QP with non-linear constraints. In chapter 5, we detail the approach we proposed based on the assumption that the contacts remain static (no sliding contact, only sticking case).

## 2.4 Conclusion

In this chapter we have review solutions for IK and ID problems. As we have seen, there is a several methods to solve these problems. Each of them having their own advantages and drawbacks. They can be compared on the following criteria: efficiency (computational speed), accuracy, robustness, generality (e.g. robot's shape, actuations), and complexity of the method (implementation). The choice of a method clearly depends on the possibilities that the robotic system offers, and the application.

In general, if a closed-form solution exists, analytic methods are always preferable. However, this is seldom the case for soft robots, and not possible for applications with unpredictable interactions. Numerical methods then often appear to be the best solution. In that respect, the main advantage of these methods is their generality and flexibility: most of them can deal with arbitrary shape of robot, and can easily integrate constraints (e.g. actuation limit, contact constraint) and solve singularity problems. However, the price to pay for this generality is often a high computational cost and high complexity of the resolution methods, which is mainly due to their iterative nature. Furthermore, they are less reliable than analytic methods since the convergence to a solution is not always guaranteed. However, as we seek for generality in this thesis, we choose to invest optimization-based method, with the objective of maintaining good computation rates.

Chapter 3

# Inverse Model of Deformable Structures

## Contents

## 3.1 Introduction

In this chapter, we detail elements of the method we use to solve the motion of soft robots without considering the environment first, and for both the forward and inverse problems. To control a soft robot, one need to be able to describe the continuum deformation of its structure. In this thesis, we choose to work with a numerical model, that is a numerical representation of the robot. The second section 3.2 of this chapter is then dedicated to this mechanical modeling. We detail the deformable models we use (elastic and hyper-elastic), the equations to solve (static and dynamic), and how we model the actuation that we chose to formulate as constraints. With all these elements set, we give in the third section 3.3, the formulation of the forward and the inverse problems, together with methods to solve them. Results from numerical and real experiments are given in the fourth section 3.4. We finally present two direct applications in the fifth section 3.5, and conclude in the sixth and last section of this chapter 3.6.

## 3.2 Modeling

The simulation of a deformable object involves to describe its internal response to external loads. This relationship between the loads and resulting deformations is given by the constitutive law of its underlying material. We briefly present constitutive laws for linear materials, and non-linear materials. In this thesis, we focus on isotropic materials, whose response to deformation is independent of the orientation that such deformation is applied in. Although, anisotropic material could be simulated as well. Afterward, we present the deformable models we usually choose to simulate soft robots, which are the beam co-rotational model (for rod-like shapes), and the co-rotational model and some non-linear models (for more general shapes). These models are defined be the constitutive law, and allow us to construct the FEM matrix that describes the internal forces of the object, usually called the stiffness matrix. A great course on 3D Deformable Solids simulations, with recommended textbooks, is given in (Sifakis & Barbic 2012). Note that we only provide the information about how we use FEM to model the robots. For a deeper presentation of FEM, please refer to (Cook et al. 2007). The understanding of the algorithms should not be impacted. The stiffness matrix takes part in the static and dynamic equations of motion, that we describe in detail. For the constraints (e.g. actuation) we use Lagrange multipliers, which involves the construction of a Jacobian matrix. We present the definition of this Jacobian for the modeling of cable actuation, pneumatic actuation, and hydraulic actuation. For some applications, it can be interesting to model mechanical parameters (e.g. Young's

modulus) as constraints that can be optimized. In that matter, we also detail the formulation of their corresponding Jacobian matrix. Let us first introduce the constitutive laws.

### 3.2.1 Constitutive laws

The simulation of a deformable object involves to describe its internal response to external loads. This relationship between the loads and resulting deformations is given by the constitutive law (constitutive equation) of its underlying material. In particular, this law gives the relation between the stress $\sigma$ and the strain $\varepsilon$. Where stress and strain respectively expresses the internal forces that the nodes exert on each other, and the measure of the deformation from a displacement of the material (see Sifakis & Barbic (2012) for more detail). This two quantities are directly linked since the internal forces in a continuum structure are caused by its deformation. This relation between stress and strain depends on the mechanical properties of the studied material, such as Young's modulus and Poisson's ratio[1], which are obtained from real experiments. As a general rule, the more the constitutive law will be precise and the more the parameters will be numerous and complex to measure.

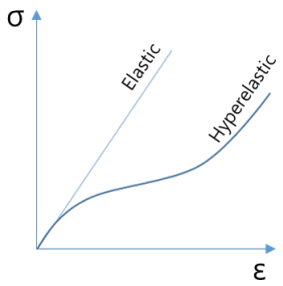


Figure 3.1: *Stress-strain graph curve illustrating the constitutive law of elastic (linear) and hyper-elastic (non-linear) materials. For small deformations, small strains, both type of materials can share a similar constitutive behavior.*

#### Linear elasticity

An elastic behavior refers to the ability of the body material to deform under external loading and to return to its initial shape when load is removed. For small deformations, most elastic materials exhibit linear elasticity. The simplest constitutive equation for linear elasticity is given by Hooke's law,

[1]Poisson's ratio is the measure of incompressibility of the material. Young's modulus is the measure of the stretch resistance.

which expresses the deformation (lengthen) of the object as proportional to the force, yielding:

$$\sigma = E \cdot \varepsilon$$

Where $E$ is a rank four tensor (see Cook et al. (2007) or Reddy (2013) for more detailed explanations) which depend on the mechanical parameters. For example, for isotropic materials, $E$ only depends on Young's modulus and Poisson's ratio.

This constitutive equation is often use to model deformable object, as its simplicity allows very fast computation. However, this law is only valid for small deformations and small rotations. When large deformations occur, Hooke's linear law is no longer a good approximation.

**Non-linear elasticity**

Many soft materials are described by non-linear elasticity theory under large deformations. The most common example of this kind of material is rubber, whose relationship between stress and strain can be defined as non-linearly elastic, isotropic, and incompressible (Erman & Mark 1997). This kind of materials can undergo large elastic deformations in order of around 100% to 700% and fully recover their initial shape when load is removed. Such large deformations are particularly observed with pressurized soft structure, built from silicone for instance.

Non-linear elasticity is more difficult to formalize and estimate. However, the strain-energy density function, $\Psi$, required for deformation of all hyper-elastic materials can be characterized as follow:

$$\sigma = \frac{\delta \Psi(\varepsilon)}{\delta \varepsilon}$$

where $\Psi$ can be a very complex function (see Sifakis & Barbic (2012)). The non-linear constitutive law of hyper-elastic materials derives from this strain-energy density function.

### 3.2.2 Deformable models

Depending on the constitutive law of the material and the geometrical representation of the robot, several possibilities exist in SOFA to model deformable materials. Here we describe the three models (or type of models) that are used in this thesis, which are the co-rotational beam model (for rod-like shapes),

and the co-rotational model or hyper-elastic models (for more general shapes). Note that, more specifically, each of these models can compute the internal forces $f$, and the FEM stiffness matrix $K$, which is involved in the static and dynamic equations presented in latter sections 3.2.3 and 3.2.3.

The relation between the internal forces and stress is then expressed by assuming that the body is in an equilibrium configuration (Sifakis & Barbic 2012), yielding:

$$f = div(\sigma) = \sum_{j=1}^{3} \frac{\partial \sigma}{\partial x_{,j}} \tag{3.1}$$

To compute the internal forces $f_e$ at each element $e$, one use the weak formulation (variational form). That is, one integrate over the domain equation (3.1) (which is called the strong form) multiplied by so-called weight-function (see Cook et al. (2007) or Reddy (1993) for more detailed explanations).

### Co-rotational beam model

Beam elements are used to model objects whose length is greater than the other transverse dimensions, or in other terms structures with a rod-like shape. Examples of objects with this geometry are plentiful in surgical robotics. Their particular nature generally leads to large geometric deformations. The approach used in SOFA is based on a co-rotational approach, to handle the important geometric non-linearity.

The co-rotational beam model, originally presented in (Dequidt et al. 2008), is based on three-dimensional beam theory. It describes the structure as a series of serially linked beams, each composed of two points at its extremities. These points are consider as rigid, and thus have six DoFs (three rotations and three translations). The corresponding stiffness matrix $K$ is a symmetric block-tridiagonal matrix. Indeed, each element is only linked to one or two other elements (see Figure 3.2).

The assumption of the co-rotational model is that the deformations remain *small* at the level of each element. Thus, a local frame is defined for each beam. The elementary stiffness matrix $K_e$, explicitly given in Cotin et al. (2005), is initially calculated in local coordinates. We note $\hat{K}_e$ the elementary stiffness matrix in local coordinates. The force $f_e$ at the level of the element is equal to:

$$f_e = \hat{K}_e(x - x_0)$$

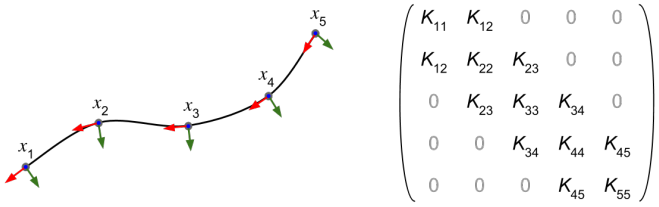


Figure 3.2: *Serial beam elements with corresponding block-tridiagonal matrix.*

where $x$ and $x_0$ reflect respectively the actual and the initial configuration of the beam, in the local frame of the beam. A transformation matrix $\Lambda$ is then defined to change the local frame of reference to a global coordinate system. This leads to the following relationship between $\hat{K}_e$ in local coordinates and $K_e$ in a global frame:

$$K_e = \Lambda^T \hat{K}_e \Lambda$$

where $\Lambda$ is a matrix obtained from the direction cosines of angles between the local and global coordinate systems. As the beam deforms, only $\Lambda$ changes and needs to be recomputed, while $\hat{K}_e$ remains constant, as long as the deformation of the beam in its local frame remains small. The global stiffness matrix $K$ is then recomputed at each time step by summing the elementary stiffness matrices $K_e$.

### Co-rotational model

For 3D deformable object with thick shapes, using purely linear model (i.e. linear strain and linear elasticity) is simpler and computationally very efficient, but it is only valid for small displacements. Indeed, these models consider a constant stiffness matrix $K$ during the entire simulation, and result in experienced unrealistic growth in volume under large rotational deformations (the reader can refer to (Müller & Gross 2004) for an example). Instead, we can use the popular co-rotational approach (Felippa 2000, Müller & Gross 2004, Sin et al. 2013). Note that this model is the original inspiration of the co-rotational beam model, and thus is quite similar to the above description. The co-rotational model for 3D deformable object provides a relatively simple way to handle large displacements and geometrical non-linearities, and offers a good compromise between computational efficiency and accuracy.

The method involves the calculation of the rotational component $R_e$ of the

element, determined via polar decomposition of the deformation gradient [2]. For each element, the local frame is represented by the matrix $R_e$, and $R_e^T$ is used to put back the element into the original global frame. The force $f_e$ at the level of the element is then equal to:

$$f_e = R_e K_e (R_e^T x - x_0)$$

where $x$ and $x_0$ represent respectively the actual configuration of the object in the global frame, and the initial configuration of the object in the local frame, and $K_e$ is the elementary stiffness matrix. The final stiffness matrix $K$ is then built from the assembly of the matrices $R_e K_e R_e^T$.

From a computational point of view, with respect to a purely linear model, co-rotational model adds the cost of a polar decomposition, and a need to solve an additional system at each time step to compute $K$. However, for most of our soft robots, using this model provides a good approximation of their global behavior at high rates, although the non-linear elastic behaviors are neglect.

### Hyper-elastic models

A first modification to the linear elastic material model to the range of non-linear deformations is given by the Saint-Venant-Kirchhoff model. The Saint-Venant Kirchhoff model is another commonly employed deformable model. It uses a non-linear strain, which ensures that the model will not result to volume growth under large rotations. This model is perhaps the simplest model for non-linear elasticity (hyper-elastic materials) because it models the stress-strain relationship with a linear function. However, Saint-Venant-Kirchhoff model has a poor resistance to forceful compression (Sifakis & Barbic 2012): the material can locally tangle and invert itself when subjected to strong compressive forces or kinematic constraints. For very large strains it is often recommended to avoid this model and instead use other non-linear models, such as neo-Hookean and Mooney-Rivlin (Misra et al. 2008), to cite only a few of the most popular ones. See (Martins et al. 2006) for a comparative study of several hyper-elastic models, applied to silicone rubber. Sofa provides all these models. However, it is known that the stability of these models is not guaranteed under very large strains. Moreover, these models generally require more parameters that can be measured only with specific equipments.

[2] The deformation function is, as its name suggests, the function that gives the relation between each material point and its respective deformed location.

### 3.2.3 Equations of motion

Thanks to FEM formulation, the constitutive laws are integrated over the elements and provide computational values of internal forces. These forces, computed at each node correspond to the numerical integration of the constitute laws over the surrounding elements. Most soft tissues are inherently viscoelastic; they have a response based on both position and velocity. If we ignore the viscosity of the material, the internal forces depend on the position of the nodes.

We now give the equations of motion we need to solve in order to compute the position and velocity of the nodes. Depending on the soft robot and the application, one can either use a quasi-static or dynamic approach. Dynamic effects on soft robots, such as vibrations, are often unwanted and avoided. In that matter, the actuations are usually done at low velocities so that the inertia forces vanish. However, when the robot has no fixed points (e.g. for locomotion), it is required to use the dynamic approach. In the following we describe the equations of motion in both cases.

#### Static equations

As mentioned above, if the robot has a fixed part, and its motion is performed at low velocity, we can ignore the dynamics and use a quasi-static approach. The configuration of the robot at a given time is then obtained by solving the static equilibrium between the internal forces of the deformable structure $f(x)$ (where $x$ is the positions of the FE nodes) and the external loads $f_{ext}$ (e.g. gravity), yielding:

$$-f(x) = f_{ext} \tag{3.2}$$

In our deformation models, the internal forces are a non-linear function. At each step $i$ of the simulation, we compute a *linearization* of $f(x)$ by applying a Taylor series expansion, leading to the following first order approximation:

$$f(x_i) = f(x_{i-1}) + \underbrace{\frac{\partial f}{\partial x}}_{K} dx \tag{3.3}$$

with $K$ the tangent stiffness matrix mentioned in section 3.2, that depends on the current position of the FE nodes, and $dx$ is the difference between consecutive positions in time $dx = x_i - x_{i-1}$. By combining equations (3.2) and (3.3) we have:

$$-Kdx = f(x_{i-1}) + f_{ext} \tag{3.4}$$

As saw above, the stiffness matrix $K$ is computed by the FEM and depends on the deformation model we use. Note that the matrix is constant during the time step, but is updated at each time step, as we consider in this work the geometrical or material non-linearities. As $f_{ext}$ are known, the only unknown here is $dx$. The solution of this problem can be obtained using the Newton-Raphson iterative method. In practice, we only solve the first iteration of the Newton-Raphson method at each time step, which provide a good solution. Note that we do not yet consider constraints into the problem.

**Dynamic equations**

When taking the dynamics effect into account, the configuration of the robot at a given time is obtained by solving the equations given by the second law of Newton:

$$Ma = f(x, v) + f_{ext} \tag{3.5}$$

where $x$ and $v$ are respectively the vector of the FEM nodes positions and velocities, $a$ is the accelerations, $M$ is the mass matrix, which can be easily computed using the FEM formulation, in particular with mass-lumping, and $f(x, v)$ represents internal forces of the deformable structure which is now a function of the positions and velocities. Note that the formulation here allows to simulate viscous internal forces.

The system of equations is now evolving in time. We use a time-stepping implicit scheme (backward Euler) to integrate the equations over time. Implicit schemes involve the inversion of a complex matrix at each time step (compared to explicit scheme), however, it gives unconditional stability which is essential in an interactive system. Given a time step $h = t_{i+1} - t_i$ and a current state $(x_i, v_i)$, the implicit scheme gives:

$$\begin{cases} Ma_{i+1} &= f(x_{i+1}, v_{i+1}) + f_{ext} \\ v_{i+1} &= v_i + ha_{i+1} \\ x_{i+1} &= x_i + hv_{i+1} \end{cases} \tag{3.6}$$

Note that here, $a_i$, $v_i$, and $x_i$ respectively correspond to the short version of $a(t_i)$, $v(t_i)$, and $x(t_i)$. We usually express this system with respect to the velocity instead of the acceleration. By defining $dv = v_{i+1} - v_i$, we obtain:

$$\begin{cases} Mdv &= hf(x_{i+1}, v_{i+1}) + hf_{ext} \\ x_{i+1} &= x_i + hv_{i+1} \end{cases} \tag{3.7}$$

In implicit scheme, the state at a certain instant cannot be explicitly expressed as a function of the state at the previous time step. Thus computing the internal forces $f$ requires the positions and velocities at the end of the time step, that are unknown. To solve this problem and obtain the final linear system, we also linearize the internal forces by applying a Taylor series expansion, leading to the following first order approximation (as done in Baraff & Witkin (1998)):

$$f(x_{i+1}, v_{i+1}) = f(x_i, v_i) + \underbrace{\frac{\partial f}{\partial x}}_{K} dx + \underbrace{\frac{\partial f}{\partial v}}_{D} dv \tag{3.8}$$

where D is called the damping matrix. In practice, we simulate viscous material using Rayleigh damping. In that case, D is defined as a linear combination of the mass and stiffness matrices $D = \alpha M + \beta K$, known as Rayleigh damping. The coefficients $\alpha$ and $\beta$ are respectively called the *Rayleigh mass* and *Rayleigh stiffness*. By combining equations (3.7) and (3.8), and using the relation $dx = x_{i+1} - x_i = hv_{i+1} = h(dv + v_i)$, equation (3.5) becomes the following linear system:

$$(M - hD - h^2K)dv = hf(x_i, v_i) + h^2Kv_i + hf_{ext} \tag{3.9}$$

This system is solved at each time-step of the simulation. Note that with the deformable models we use, the matrix $K$ continuously evolves due to geometrical and/or material non-linearities. Many different solvers can be employed. The solution of this equation $dv$, is then used to update the Euler scheme: $v_{i+1} = v_i + dv$ and $x_{i+1} = x_i + hv_{i+1}$.

**Notation and matrices properties**

For more clarity, and to keep the approach generic in the remainder of the manuscript we will note:

*static* :

$$\underbrace{-K}_{A}\underbrace{dx}_{x} = \underbrace{f(x_{i-1}) + f_{ext}}_{b}$$

*dynamic* :

$$\underbrace{(M - hD - h^2K)}_{A}\underbrace{dv}_{x} = \underbrace{hf(x_i, v_i) + h^2Kv_i + hf_{ext}}_{b}$$

Note that the FE matrices $K$, $D$, and $M$ are symmetric, positive definite and sparse through construction. The matrix $A$ of the system is then invertible. For the quasi-static approach, at least six independent directions must be fixed to remove the rigid motion of the solid. For the co-rotational beam model we generally fix the first node in position and rotation. For the other models, we fix at least more than three points in position.

### 3.2.4 Constraints

The generation of movements of soft robots is coming from a change in the balance of forces induced by the actuation. In our framework, we handle the actuation by defining specific constraints with Lagrange multipliers on the boundary conditions of the deformable models. Several types of actuation are considered in this thesis: cable, pneumatic, and hydraulic actuators, and also mechanical parameters as a method to modulate the actuation.

The forces of the actuation are added to the external forces in equation (3.9). Considering an actuation $a$, it yields to:

$$Ax = b + H_a^T \lambda_a \tag{3.10}$$

where $H_a^T$ can be seen as the Jacobian matrix of the actuation, and $\lambda_a$ represents the effort of the actuation. More specifically the matrix $H_a^T$ contains the direction of the actuator forces on the set of involved FE nodes. It is a column matrix of the size $(n, n_a)$, where $n$ is the number of DoFs of the FE mesh, and $n_a$ is the number of actuation variables. $\lambda_a$ is then a vector, whose each line corresponds to a single actuation. Note that, for simplification we consider here the static case. In the dynamic case we have $hH_a^T\lambda_a$ instead in the equation (3.10). In the following, we detail $H_a$ construction for each actuation type.

#### Cable actuation

For cable actuation, a single constraint is set for each cable. It corresponds to the intensity of the force that the cable exerts on the soft structure. It is assumed that the cables are inextensible so that their length is directly linked to the actuator motion. Given the pulling point position $p_{pull}$ (where the actuator pulls the cable), the length is linked to the configuration of the soft robot. For each cable, we can define the function $\delta_a(x) : R^n \rightarrow R$ that provides its length, given the positions $x$ of the mesh nodes. The cable motion could be limited

by physical stops on the actuators[3]. To satisfy this constraint we can impose $\delta_a(x) \in [\delta_{min}\ ,\ \delta_{max}]$.

In the most simple cases, the cable is just attached to one node $s$ of the robot mesh and creates a force oriented towards its attachment position. In that case, the length is computed using an Euclidean norm: $\delta_a(x) = \|x_s - p_{pull}\|$, with $x_s$ the position of the pulled point. However, like many soft robots, we use a more complex path for the cable inside the robot (see Figure 3.3), in order to create more complex and possibly antagonistic motions.

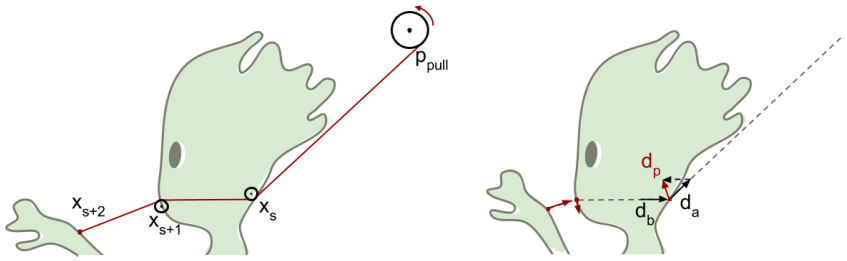


Figure 3.3: *Cable actuation of a puppet hand. Representation of quantities of equation (3.11). The cable is fixed to the hand of the puppet and is passing through its head allowing to bring the hand to the mouth.*

In practice, we place a flexible tube inside the robot soft structure that allows the cable to slip with low friction. To represent, in the model, the additional rigidity created by these tubes, we use a model of stiff springs in the direction of the tubes. In case the cable path goes inside the robot, the length of the cable depends on the motion of several nodes $s$, $s+1$, ..., $s+N$ of the mesh. It is computed as a sum of Euclidean norms:

$$\delta_a(x) = \|x_s - p_{pull}\| + \sum_{i=s}^{N} \|x_{i+1} - x_i\| \tag{3.11}$$

See Figure 3.3 for an illustration of these quantities.

In practice, it is not easy to have the nodes of the FE mesh at the same location as the attachment points and the passing points of the cable. If these points are defined in the middle of an element, we use the mapping strategy of SOFA. That is, the interpolation functions of the element $\phi_k(\alpha, \beta, \gamma)$, evaluated with the barycentric coordinates $(\alpha, \beta, \gamma)$ of the point position inside the element: $x_s = \sum_k \phi_k(\alpha, \beta, \gamma) x_k$.

[3]In most of our examples, the course of the actuator does not exceed 300mm (due to the way we build our prototypes).

The matrix $H_a$ is built as follow. Let us suppose that the points are numbered starting from the extremity where the cable is attached to the actuator. At each point $s+i$, $i \in \{0, 1, .., N\}$, we take the direction of the cable *before*, $d_b$ and *after*, $d_a$:

$$d_b = \frac{x_{s+i} - x_{s+i+1}}{\|x_{s+i} - x_{s+i+1}\|}, d_a = \frac{x_{s+i-1} - x_{s+i}}{\|x_{s+i-1} - x_{s+i}\|}$$

To obtain the direction of the constraint that is applied on the point, we use $d_p = d_a - d_b$. Note that the direction of the final point ($x_{s+2}$ on the Figure 3.3) is equal to $d_a$ as $d_b$ is not defined. These constraint directions are then mapped on the mesh nodes using the barycentric interpolation:

$$H_a = \begin{bmatrix} \dots & \phi_k(\alpha, \beta, \gamma)\ d_p^T & \dots \end{bmatrix}$$

**Pneumatic actuation**

Inflating cavities thanks to pressured air (see Figure 3.4) is commonly used by the soft robotics community. For this type of actuation, the effort $\lambda_a$ represents the pressure exerted on the cavity wall. Like for cable actuation, a single constraint is set for each pneumatic actuator. Indeed, the pressure inside the cavity is uniform over the wall.

The corresponding Jacobian matrix represents how this internal pressure will create forces on the nodes that are placed on the walls. The matrix is built as follow. For each triangle $t$ of the cavity wall, we compute its area $a_t$ and its normal direction $n_t$. The multiplication of these results by the pressure gives the vector of force applied by the pneumatic actuation on the triangle. Thus, we divide the triangle area by 3 to distribute the contribution to its nodes (see Figure 3.4). We sum up the results of each triangle in the corresponding column of $H_a$:

$$(H_a)_i = \sum_{t \in S, i \in t} \frac{a_t}{3} n_t$$

where $S$ is the set of the cavity triangles, and $i$ iterate over the nodes of the triangles.

In the particular case of a pneumatic actuation, $\lambda_a$ provides the difference between the pressure inside the cavity and the atmospheric pressure. Usually, pneumatic actuators only provide positive pressure so $\lambda_a \geq 0$. However, it is also possible to create both negative and positive pressure using vacuum

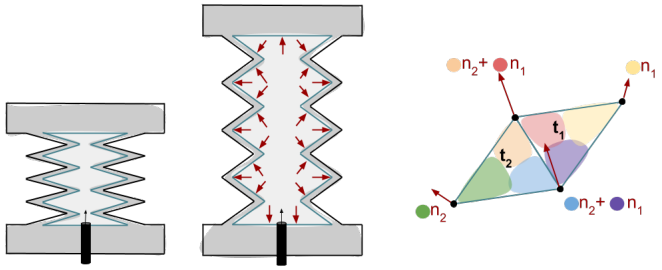


Figure 3.4: *Pneumatic actuation on an accordion model. Left: accordion at rest. Middle: accordion when inflated. Right: method of pressure distribution on the cavity wall vertices.*

actuation. In that case, the only constraint on the unknown value of $\lambda_a$ is a maximal or minimal limit of pressure that can be achieved by the actuator.

For each pressurized cavity, the function $\delta_a(x) : R^n \rightarrow R$ provides its volume, given the positions $x$ of the mesh nodes. A constraint on the volume growth can also being set by imposing $\delta_a(x) \in [\delta_{min} \ , \ \delta_{max}]$.

### Hydraulic actuation

For hydraulic actuation it is assumed that the cavity is entirely filled with the fluid, i.e. there is no air inside the cavity. Hydraulic actuation is particular as it adds two constraints $H_p^T \lambda_p$ and $H_w^T \lambda_w$, corresponding to a pressure term and a fluid weight term respectively. The pressure term is equal to that of the pneumatic actuation. The fluid weight term is a bit more complex to obtain as it requires an accurate computation of the fluid weight distribution over the cavity wall. In practice, these two terms are coupled when one activates the actuator by adding or removing the liquid. Consequently, both terms must be coupled in a single constraint defining the behavior of the hydraulic actuation, and the function governing this coupling has a highly non-linear nature. Our goal is to keep the same formulation $H_a^T \lambda_a$ (see equation (3.10)) for the total contribution of hydraulic actuators.

The matrix $H_w^T$ should contain the direction of the weight distribution with respect to the volume change. However this is complex since it depends on the cavity shape deformation. This is in fact a highly non-linear relation, but it can be linearized around the current configuration, assuming small weight variations. This assumption is valid since the soft robot control pipeline is recomputed every few milliseconds, thus changes between iterations are indeed small.

For a given configuration, we compute the weight distribution per node, $w_i$ (see Figure 3.5) and set:

$$(H_w)_i = \frac{w_i}{\vartheta}$$

with $\vartheta$ the current cavity volume. $\lambda_w$ represents then the cavity volume change.

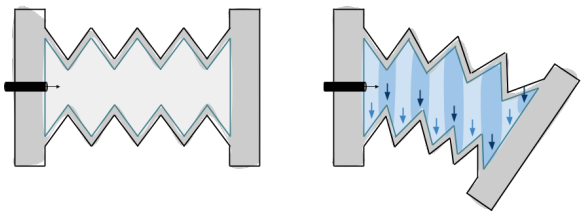

Figure 3.5: *Pneumatic (left) and hydraulic (right) actuation on an accordion model. The hydraulic actuation is showing with a distribution of the liquid weight over the cavity wall (mesh-on-grid discretization).*

Lastly, we need to merge both pressure and weight terms. The nodal displacements due to the pressure term is given by $\Delta x = A^{-1}H_p^T\lambda_p$. These displacements can be regarded as extrusion lengths for a given surface shape with area $a$, extruded along the surface normal direction, and would thus produce a volume change $a\Delta x$. Since this surface information is already stored in $H_p$, we linearize this relation as $\lambda_w = H_pA^{-1}H_p^T\lambda_p$ which couples both terms. Therefore, we merge both terms into a single hydraulic constraint $\lambda_a$ with:

$$H_a^T = H_p^T + H_w^T H_p A^{-1} H_p^T$$

Again, this linearization remains accurate only for small changes from a given configuration and must be updated every iteration.

For the weight distribution. Although it is an easy task on analytical shapes, the typical piecewise FE models use unstructured triangle or quad meshes for their geometrical description. The exact computation of the weight distribution in those cases becomes a highly complex geometrical problem. Instead, we address this as a *mesh-on-grid* discretization problem (as illustrated in Figure 3.5). The computation of the weight distribution can be very expensive. In (Rodríguez et al. 2017), we propose this formulation of the Jacobian matrix of hydraulic actuation. This project was done in collaboration with A. Rodriguez (who visited our group during six months). In the same paper, he proposed parallel approach on GPU to speed up the computation of the weight distribution.

#### Mechanical parameters

We can also set constraints on mechanical parameters, which influence the computation of the internal forces. We propose in (Coevoet et al. 2014) to set the Young Modulus as a Lagrange multiplier, and find its value by optimization. We note the corresponding Lagrangian multiplier $\lambda_p$, and the Jacobian matrix $H_p$. To compute the Jacobian matrix, we use a local derivation of the internal force by the parameter $p$, yielding:

$$H_p^T = \frac{f(x, p + dp) - f(x, p)}{dp}$$

where f is the internal forces, and $dp$ is the variation of the parameter. To keep the validity of the local derivation over a step $i$, we can set $-\epsilon \leq \boldsymbol{\lambda}_p \leq \epsilon$. Note that this multiplier $\lambda_p$ will be solved in the same manner than $\lambda_a$ for actuators.

### 3.2.5 Effector and task space definition

We can control the motion of a soft robot either by controlling specific points if its structure, or by controlling computed points, such as its barycenter or its center of mass. We will call *effector*, each particular point on the robot that we wish to control. The task space is the positions and/or orientations accessible by the effectors. This space depends on the design of the robot (i.e. geometry, actuation), its boundaries are entirely defined by the actuation limits (stops). However, obstacles can also reduce this space.

#### Point effector

We can consider multiple effectors, and specify a constraint in the task space for these points. It is particularly useful to simulate the inverse model of the robot. It is also possible to add a load to these points in both the forward and inverse model. The function $\delta_e(x) : R^n \rightarrow R^3$ measures the shift along $x, y$ and $z$ directions between this point and the desired position $x_{des}$ or trajectory, $\delta_e(x) = x - x_{des}$. Consequently, the Jacobian matrix is given by:

$$H_e^T = [0 \dots 0 \; I \; 0 \dots 0]^T$$

whit $I$ is a $3 \times 3$ identity matrix at the location of the controlled point. Sometimes, it is interesting to define several effector points and/or to control particular directions ($x, y$ but not $z$ for example). The principle remains the same, it only changes the size of $\delta_e$ and identity matrix I.

We can also define loads $\lambda_e$ which are applied on the effector point(s) along the defined direction. Usually these loads are constant and can be easily handled in direct or inverse model using equality constraints. If no load is applied on the effector point(s), then $\lambda_e = 0$.

In the case of rigid points, such as for the beam model, one can also control points in rotation. Similarly to the control in position, one can decide to control only one or more rotations.

**Barycenter and center of mass as effector**

In some particular cases, such as locomotion for instance, it can be interesting to solve the position of the barycenter or center of mass of the soft robot. The function $\delta_e(x) : R^n \rightarrow R^3$ measures then the shift along $x, y$ and $z$ directions between this point and its desired position $x_{des}$ or trajectory, $\delta_e(x) = x - x_{des}$. The Jacobian matrices are given by:

$$Barycenter : (H_e)_i = \frac{1}{\#nodes}$$
$$Center\ of\ mass : (H_e)_i = \frac{1}{w}$$

with $\#nodes$ the total number of nodes, and $w$ the mass of the object. Typically no load is applied on these points, and $\lambda_e = 0$.

## 3.3 Solving the constraints

In this section we detail the algorithm formulations in case of forward and inverse problems. We start by presenting the common structure of both problems based on a free motion step. We finally detail the case of the forward kinematics and dynamics problem, following by IK and ID formulation and resolution.

### 3.3.1 Free motion

From equation (3.10) in dynamics or in quasi-statics, the equation has two unknowns: first, $x$ which provides the motion of the DoFs, and second $\lambda_a$, which is the intensity of the actuators loads. Consequently, the resolution will be executed in a two steps fashion.

The first step consists in obtaining a $free$ configuration $x_{free}$ of the robot that is found by solving equation (3.9) while considering that there is no actuation applied to the deformable structure, that is with $\lambda_a = 0$:

$$Ax^{free} = b \tag{3.12}$$

After solving this linear equations, given this new $free$ position $x_{free}$ for all the nodes of the mesh, we can evaluate the values of $\delta_i^{free} = \delta_i(x^{free})$ with $i \in \{a, e\}$, defined in the previous section 3.2.4.

The second step is based on an optimization process that provides the value of $\lambda_a$. The approach relies on an optimization process and its output is the value of the Lagrange multipliers. The size of matrix $A$ is often very large so an optimization in the motion space would be computationally very expensive. To perform this optimization in real-time, we use the well-known Schur complement:

$$\delta_i = \underbrace{\left[H_i A^{-1} H_j^T\right]}_{W_{ij}} \lambda_j + \delta_i^{free} \tag{3.13}$$

with $i$ and $j \in \{a, e\}$. The above quantities are illustrated in Figure 3.6. Equation (3.13) corresponds to the static case, for the dynamic case we have $W_{ij} = h^2 H_i A^{-1} H_j^T$. Note that in that case we use a $LDL^T$ factorization of the matrix $A$ and do not directly compute its inverse. The physical meaning of this Schur complement is central in the method: $W_{ij}$ provides a measure of the instantaneous mechanical coupling between the boundary conditions $i$ and $j$, whether they correspond to an effector or an actuator. In practice, this method allows to perform the optimization with the smallest possible number of equations.

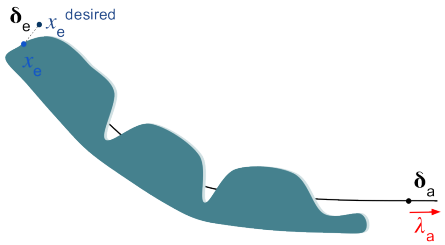


Figure 3.6: *Illustration of the quantities of equation (3.13) for a deformable robot actuated with one cable.* $x_e^{desired}$ *is the desired position for the controlled point* $x_e$.

It should be emphasized that one of the main difficulties is to compute $W_{ij}$ fast enough. No pre-computation is possible since the value changes at each iteration. But this type of problem is frequent when solving friction contact

on deformable objects, thus several strategies are already implemented in SOFA (Faure et al. 2012).

At this stage, the value of $\lambda_a$ is either solved with the optimization process we describe in following section 3.3.2 for the inverse problem, or set manually for the forward problem. With this value we can compute the final configuration of the soft robot, at the end of each time step using:

$$x = x^{free} + A^{-1} H_a^T \lambda_a \tag{3.14}$$

which provides the solution to equation (3.10). This solution corresponds to the static case, for the dynamic case we have $hA^{-1}H_a^T$. In the case of the forward problem, the inputs are the actuator values, and can be either $\delta_a$ or $\lambda_a$.

### 3.3.2 Inverse problem

In the case of the IP, the inputs are the desired position of effectors and the outputs are the force $\lambda_a$ or the motion $\delta_a$ that needs to be applied on the actuators in order to minimize the distance with the effectors position.

#### Jacobian

As explained above, using the operator $W_{ea}$, we can get a measure of the mechanical coupling between effectors and actuators, and with $W_{aa}$, the coupling between actuators. On a given configuration, $W_{ea}$ provides a linearized relationship between the variation of displacement $\Delta\delta_e$ created on the effectors and the variation of the effort $\Delta\lambda_a$ on the actuators. To get a direct kinematic link between actuators and effectors, we need to account for the mechanical coupling that can exist between actuators. This coupling is captured by $W_{aa}$ that can be inverted if actuators are defined on independent DoFs. Consequently, we can get a kinematic link by rewriting equation (3.13):

$$\Delta\delta_e = W_{ea} W_{aa}^{-1} \Delta\delta_a \tag{3.15}$$

This relationship provides (in the most condensed way) the displacement of the effector given the displacements of the actuators. We found that matrix $W_{ea}W_{aa}^{-1}$ is equivalent to a Jacobian matrix for a standard, rigid robot. This corresponds to a local linearization provided by the FEM model on a given configuration and this relationship is only valid for *small variations* of $\Delta\delta_a$, and in contactless cases.

### Optimization

A first solution to inverse the model of the robot was provided in (Duriez 2013) using an adaptation of a Gauss-Seidel solver that was used originally to solve contact constraints. When dealing with constraints on actuation (such as limit on cable displacement) it is more convenient to formulate the problem as an optimization program, as we propose in (Coevoet et al. 2014, 2015). Note that, it is $\lambda_a$ that is found by optimization, and $\delta_a$ can be obtained using equations (3.13). Let consider an effector $e$. The optimization consists in reducing the norm of $\delta_e$, which actually measures the shift between the effector and its desired position. Thus, computing $min(\frac{1}{2}\delta_e^T\delta_e)$ can be done by setting the following QP problem:

$$\min_{\lambda_a} \left\| W_{ea}\lambda_a + \delta_e^{\text{free}} \right\|^2 \tag{3.16}$$

$$\text{s.t. } \delta_{min} \leq \delta_a = W_{aa}\lambda_a + \delta_a^{free} \leq \delta_{max} \tag{3.17}$$

$$\lambda_a \geq 0$$

where equations (3.17) are respectively constraints on actuators course, and constraints on actuators effort (here for unilateral effort). The use of a minimization allow us to easily set these constraints to the problem. Indeed, most of the actuations are limited by the hardware, such as the cable displacement, the pneumatic pressure (e.g. not allowed negative pressure), or the amount of liquid in the case of hydraulic actuation. Other constraints can also be *physical*, such that the fact that a cable can not push. All these constraints are added to the minimization problem. The use of a minimization also allows us to find a solution even when the desired position is out of the workspace of the robot. In such a case, the algorithm will find the point that minimizes the distance with the desired position while respecting the limits introduced for the stroke of the actuators.

To solve this QP, numerous open-source solver are available. We use as a third-part the solver provided by the qpOASES library (Ferreau et al. 2014). Note that the size of the optimization is equal to the number of actuator variables, and this size is often small. From our experimentations for soft robots with a number of actuators up to 12, and a number of constraints up to 24 (generally one constraint for unilateral effort and another one to limit the course for each actuator), the solver converge in less than 10 iterations and less than $0.06ms$ [4]. Again, this solver provides the values of $\lambda_a$, and is followed by the computation of the final configuration using equation (3.14).

[4] On a laptop with an i7 Intel processor 2.90GHz ×8

### Well posed problem

The matrix of the QP, $W_{ea}^T W_{ea}$, is symmetric through construction. However, if the number of actuators is equal or less than the size of the effector space (number of controlled points $\times$ controlled directions), the matrix is also positive-definite. In such a case, the solution of the minimization is unique. In the opposite case, i.e. when the number of actuators is greater than the DoFs of the effector points, the matrix of the QP is only semi-positive, and the solution could be non-unique. In such a case, some QP algorithms are able to find one solution among all possible solutions (Sha et al. 2002). However, the solver may swing from one solution to another, entering in an unstable state.

To avoid that, we add to the objective function, an expression of the actuators mechanical work of the actuator forces $E = \Delta\delta_a \lambda_a$, with $\Delta\delta_a = \delta_a - \delta_a^{free}$ the displacement of the actuators during a time step. $E$ is linked to the mechanical energy of the robot deformation. The objective becomes $(\frac{1}{2}\delta_e^T \delta_e + \epsilon E)$, and the QP matrix is regularized. The coefficient $\epsilon$ has to be chosen sufficiently small to keep a good accuracy on the effectors motion. Indeed, if this number is to large, the optimization will tend to under-evaluate the actuation forces, and then keep the effectors at distance to the targets, even though a perfect match is possible. In practice we choose $\epsilon = 1e^{-3}||W_{ea}^T W_{ea}||_\infty / ||W_{aa}||_\infty$ (with the norm $||.||_\infty$ being the maximum absolute row sum of the matrix).

A unique solution of the problem can then be found, without a significant impact on the quality of the solution. In the rest of the manuscript, where we add contact into the process, we omit this damping term , for the sake of clarity. It is straightforward to extend the developments to incorporate it.

## 3.4 Experiments and results

As mentioned, SOFA offers several deformable models to simulate our soft robots (e.g. co-rotational models, and hyper-elastic models). A careful attention should be taken with the modeling (mesh, models, constraints etc.), as it has an impact on the accuracy of the simulation. In this section we conduct several experiments and discuss some practices to obtain good match between the simulation and the reality: first, we discuss the effect of the discretization on the simulation, second, we show results of different deformable models (linear and non-linear elasticity) on few actuated soft structures. Finally we give results of our control methods on several numerical and real experiments. Note that, when not specified, the deformable model used is the co-rotational model.

### 3.4.1 Discretization

The FEM involves a spacial discretization of the continuum, that is the creation of a *mesh* giving a discrete representation of the object. The meshing step is important, as it has a direct incidence on the accuracy of the solution, but also on the size of the system to solve, and accordingly the computational expense of the resolution. Indeed, the more nodes the mesh will have, the more accurate the FE solution will be. However, to allow fast computation, it is often required to leverage this number of nodes. It is advised to have enough points in the parts of the structure where it is known that big deformations will occur, but not too much points where the deformations will be low. Such non-uniform mesh (in terms of element size) can be difficult to get in practice [5], and a uniform meshing is often used instead.

Depending on the geometry of the object, the generation of the mesh can be more or less difficult, even when considering uniform meshing. While it is easy for thin structures, which deformations can be described with 1D or 2D meshes, it is often more complex for thick structures. Typically, a tetrahedral or hexahedral representation is needed, and the elements should fit the surface while being conform and structured (see Lo (2014)). The generation of 3D meshes is still an active domain of research (Alliez et al. 2005, Lo 2014). However powerful tools already exist, such as the open-source softwares CGAL and GMSH, to cite only those we used along this thesis. Note that a plugin based on the CGAL library is available with SOFA.

In Figure 3.7 and Table 3.1, we give results of different mesh resolutions on a soft structure actuated with pressure in its central cavity. The geometry of the object is a bit complex, first because the surface that deforms is large and thus require a large number of nodes, and second because the shape of the cavity is hard to satisfy without a lot of nodes. Also, we can see from the numerical experiment in Figure 3.14, that the global behavior of the structure is very sensitive to mesh variation. For this kind of example, in order to get fast computation with good accuracy, the best solution is to use model order reduction techniques like proposed in (Goury & Duriez 2018) (see results in Figure 3.7 and Table 3.1). In Figure 3.8 and Table 3.2 we give an example, when this time the geometry of the soft object allows for simulation at high rates with good accuracy (i.e. when considering as solution results from the mesh with the highest resolution).

[5] Tools in SOFA are available to generate such meshes. For instance, with the use of 3D images to represent different regions and CGAL library to generate the mesh, it is possible to build a mesh with different element size per regions.

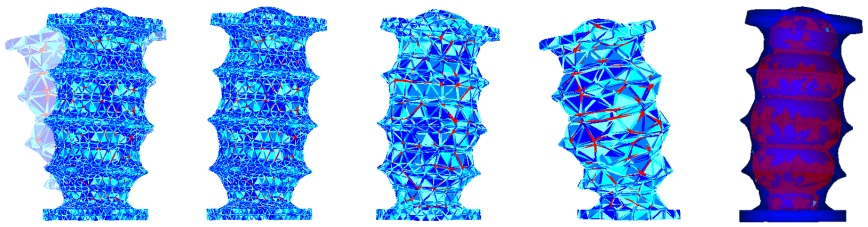

Figure 3.7: *Different meshes effect on the behavior of a pressurized accordion. Same volume growth (700%) is applied to each model. From left to right: superposition of meshes, mesh #1 has a high resolution, mesh #2 and mesh #3 which have less elements and less points, mesh #4 which is a result of model order reduction (based on mesh #1) proposed in (Goury & Duriez 2018). Corresponding numerical quantities are given in Table 3.1.*

| Mesh | #DoFs | #Nodes | #Elem. | Pressure | Time | Error |
|---|---|---|---|---|---|---|
| #1 | 12678 | 4226 | 14508 | 1.55 *bar* | 858.4 *ms* | 0.00 *mm* |
| #2 | 4518 | 1509 | 4907 | 1.81 *bar* | 142.6 *ms* | 1.02 *mm* |
| #3 | 1857 | 619 | 1975 | 2.11 *bar* | 35.36 *ms* | 20.94 *mm* |
| #4 | - | - | - | 1.57 *bar* | 5.43 *ms* | 2.06 *mm* |

Table 3.1: *Numerical results of simulations from Figure 3.7. The object is of* 50*mm height. The number of DoFs, number of nodes, number of elements, pressure in the cavity, time to compute one step of the complete simulation (without rendering), and the end-effector position error when considering the mesh #1 as the reference solution.*

Note that using low resolution may also introduce an artificial rigidity in the simulated structure. For instance, in the finger example, for a same cable displacement it requires more force to pull the cable when the mesh has less points (same for pressure example). This effect has no big impact on the control of cable-driven robots, since we usually command cables in displacement. However for pneumatic actuation it requires a calibration step, since the control are usually done in pressure and not volume growth. Note that this artificial rigidity may also be compensate by decreasing the elasticity parameter in the simulation (e.g. Young's modulus).

To conclude, a careful attention should be given to the mesh generation: (1) we should be careful that there are enough points to describe the deformation, (2) but not too much to allow for fast simulation, (3) when working with small number of points, an artificial rigidity is introduced and can be taking care of. A lot of great and powerful softwares are available, and allow to tune

meshes to obtain good accuracy with high computation rates [6]. Finally, when it seems not possible to satisfy this two criteria, one can still use model reduction methods (Goury & Duriez 2018).

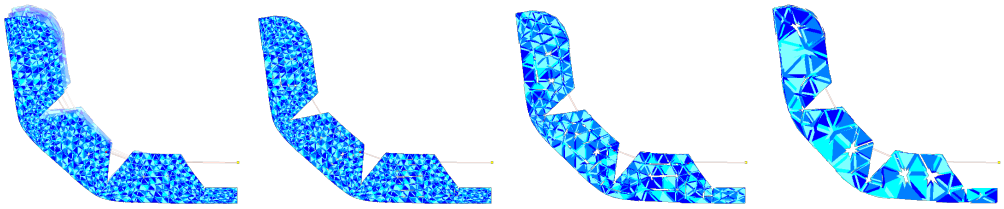

Figure 3.8: *Different meshes effect on the behavior of a cable-driven finger similar to (Manti et al. 2015). Same cable displacement (30mm) is applied in each case. From left to right: superposition of the three meshes, mesh #1 has high resolution, mesh #2 and mesh #3 have less elements and less points. Corresponding numerical quantities are given in Table 3.2.*

| Mesh | #DoFs | #Nodes | #Elem. | Force | Time | Error |
|---|---|---|---|---|---|---|
| #1 | 16227 | 5409 | 24768 | 0.327 $N$ | 907.6 $ms$ | 0.00 $mm$ |
| #2 | 2553 | 851 | 3096 | 0.348 $N$ | 36.4 $ms$ | 3.96 $mm$ |
| #3 | 447 | 158 | 389 | 0.465 $N$ | 3.5 $ms$ | 7.52 $mm$ |

Table 3.2: *Numerical results of simulations from Figure 3.8. The object has the size of* $100 \times 15 \times 15mm$. *Number of DoFs, number of nodes, number of elements, force exerted by the cable, time to compute one step of the complete simulation (without rendering), and the end-effector position error when considering the mesh #1 as the reference solution.*

### 3.4.2 Deformable models

Depending on the range of deformation that we want the robot to perform, we can choose between several deformable models. In this section, we propose a simple comparison between elastic and hyper-elastic models, on the two same examples that were used in the latter section. In Figures 3.9 and 3.11, we visually compare the co-rotational, Mooney-Rivlin, Saint-Venant Kirchhoff and Neo-Hookean models. For both accordion and finger examples, the same mesh is used to compare each model.

In the accordion example, hyper-elastic models show different behaviors. This may be due to the fact that the corresponding parameters were compute from

[6] Computation times from Table 3.1 and 3.2 were performed on a laptop with an i7 Intel processor 2.90GHz ×8

the Young's modulus and Poisson's ratio used for the co-rotational model, instead of real measurement. For large deformation, when using the simulation as a tool for design, we advise to use (in SOFA) either Saint-Venant Kirchhoff or Neo-Hookean models. When the design step is finished and the prototype is built, an additional time may be required to choose the right variables (such as the mesh resolution, model, mechanical parameters and constraints) that make the simulation match the reality.

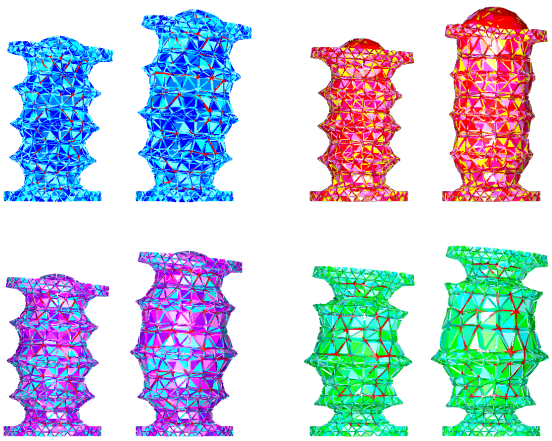

Figure 3.9: *Difference between co-rotational (blue), Mooney-Rivlin (green), Saint-Venant Kirchhoff (red) and Neo-Hookean (purple) models on the accordion example. The results of each model are superposed. Imposing the same volume growth.*

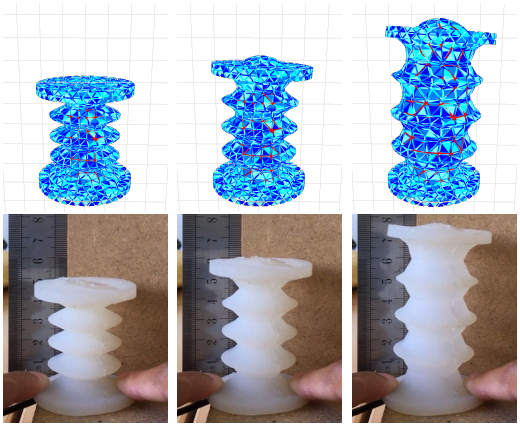

Figure 3.10: *Co-rotational model and real structure. Visualization of the deformation for same elongation.*

In the finger example we see that the difference between linear elasticity and hyper-elasticity (Saint-Venant Kirchhoff and Neo-Hookean) are low, while

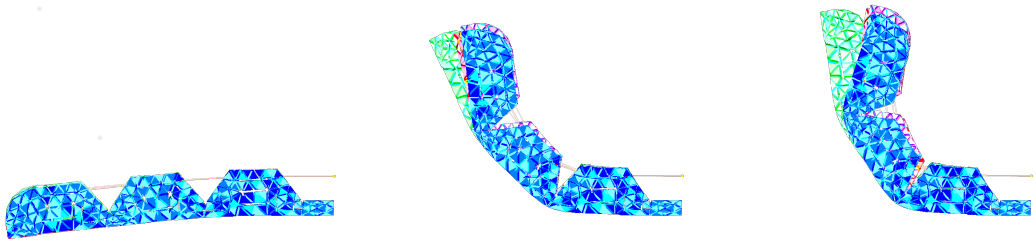

Figure 3.11: *Difference between co-rotational (blue), Mooney-Rivlin (green), Saint-Venant Kirchhoff (red) and Neo-Hookean (purple) models on the finger example. The results of each model are superposed. (left) The object is only subject to gravity, no cable force is applied, the models are similar. (middle) Imposing a displacement of* $30mm$*. (right) Imposing a displacement of* $39mm$*.*

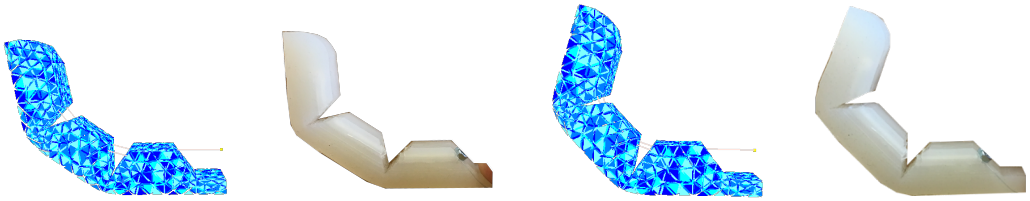

Figure 3.12: *Co-rotational model and real structure. (left) Imposing a displacement of* $30mm$*. (right) Imposing a displacement of* $39mm$*.*

Mooney-Rivlin model gives very different results. Note that this difference with the Mooney-Rivlin example may come from the corresponding implementation of the model in SOFA, or the chosen parameters. In Figure 3.12, we show a comparison between a real prototype made of silicone, and the corresponding co-rotational model correctly parameterized.

To conclude, depending on the geometry of the robot, and the range of deformation we want to apply, a careful attention should be given to the model we use. While in most of our experiment with built soft robots the co-rotational model was accurate enough, there is no restriction to use our algorithms with hyper-elastic models.

### 3.4.3 Motion control

In this section we give results on both numerical and real experiments of our controllers. As a preliminary results, for the numerical experiments, IK is confronted to the forward resolution. However, the main sources of error when controlling a real robot come from the model approximations. In that matter,

results from real experiments on a soft trunk are presented. The performances of the method, with respect to computation time, are discussed in section 3.5, where we present direct applications of our controllers.

### Numerical experiments

We present here, a validation of the approach using numerical examples. In the following, all the calculated errors are errors of the optimization results with respect to the forward resolution. In that matter, the same modeling (i.e. mesh, deformable model, mechanical parameters etc.) is used in compared simulations. The experimental protocol is the following: first, we create arbitrary deformation on a deformable object (in our case an accordion and the *Stanford Bunny* from the Stanford Computer Graphics Laboratory) by modifying boundary conditions (cable or pressure) or model parameters (Young's modulus). Second, relevant points of the model are chosen and their position is stored when equilibrium is reached. Finally, the simulation is restarted without the deformation and the selected points are given as target to our inverse approach (leading to a perfect match since we add a perfect correspondence between the points in the undeformed or deformed states). We then compare the difference between the actual values of boundary conditions, or parameters that have been used and the ones estimated through the inverse method.

**Cable actuation.** The first numerical experiment is conducted on a soft cable-driven robot. By applying different displacements of the three cables of a soft accordion structure, on a forward simulation, we extract the resulting position of its end-effector. Afterward, we make our approach optimize the cable displacement that solve the end-effector position, the target being the position extracting from the forward simulation (see Figures 3.13). Our method estimate the displacements with less than 1% error. The values are listed in the Table 3.3.

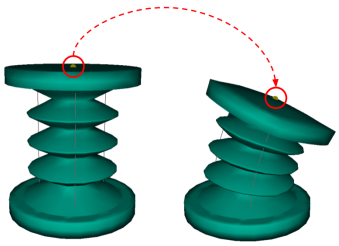

Figure 3.13: *Cable displacement estimation. (left) soft accordion at rest (right) inverse simulation by registration of the end effector (highlighted in red).*

| Cable | Applied Disp. | Estimated Disp. |
|---|---|---|
| #1 | 5 $mm$ | 5.07 $mm$ |
| #2 | 10 $mm$ | 10.00 $mm$ |
| #3 | 15 $mm$ | 14.97 $mm$ |

Table 3.3: *Performance of our approach to retrieve the three cables displacement applied to the deformable model of Image 3.13. The error in cables displacement is below* 1%.

**Pressure actuation.** A second experiment is conducted on the Stanford Bunny, which is actuated in this example with pressure in four cavities placed in its structure. We apply different pressures in cavities in a forward simulation, and the inverse problem objective is to estimate the pressures that lead to the deformation (Figure 3.14). Again, our approach estimates the corresponding pressures with less than 1% error. The values are listed in the Table 3.4.

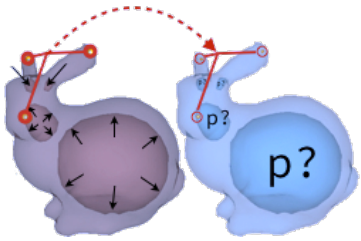


Figure 3.14: *Pressure estimation. (left) forward simulation by setting pressures in four different cavities (right) inverse simulation by registration of three points.*

| Cavity | Applied Pressure | Estimated Pressure |
|---|---|---|
| #1 | 25 $kPa$ | 25.1 $kPa$ |
| #2 | -30 $kPa$ | -29.9 $kPa$ |
| #3 | -35 $kPa$ | -35.1 $kPa$ |
| #4 | 20 $kPa$ | 20 $kPa$ |

Table 3.4: *Performance of our approach to retrieve pressures (either positive or negative) applied in cavities of the deformable model of Figure 3.14. Error in each cavity is below* 1%.

**Young's modulus optimization.** The final experiment follows the same methodology and is based on a heterogeneous material composed of three different stiffness (then three different Young's moduli to estimate). In this experiment, the deformation is induced by the softness of the model and gravity forces (Figure 3.15). Our approach estimates the three Young's moduli with less than 1% of error (see the Table 3.5).

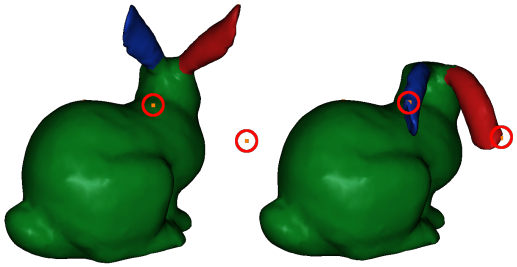

Figure 3.15: *Numerical validation. Young's modulus estimation under known gravity forces: (left) target points (highlighted in red) after setting three different Young's moduli (one color by Young's modulus), (right) the resulting deformation once the modulus have been estimated.*

Note that, in the above examples, the errors were expected to be very low. These low errors will not prevent the controller from providing solutions that do

| Bunny | Real YM | Initial YM | Estimated YM |
|---|---|---|---|
| Right Ear | 1 $kPa$ | 10 $kPa$ | 0.928 $kPa$ |
| Left Ear | 5 $kPa$ | 10 $kPa$ | 4.852 $kPa$ |
| Body | 2 $kPa$ | 10 $kPa$ | 2.027 $kPa$ |

Table 3.5: *Performance of our approach to retrieve Young's Moduli applied to different parts of our deformable model. At initialization, the Young's Moduli are set with an arbitrary value and with a perfect registration, our approach estimates the Young's Moduli with less than* 3% *of error.*

not perfectly match the reality. Indeed, most errors between the optimization and the real robot come from errors in the modeling.

### Real experiments

With a cable-driven soft trunk, we demonstrate the motion control on a real soft robot. The robot is made of silicone, and is actuated with eight cables disposed each 90 degrees around its longitudinal direction. Four cables actuate a first section (from extremity to middle) while the other four go through the entire trunk, allowing it to perform a S-shape (see Figure 3.17). As already mentioned, to avoid friction between cables and silicone, we place flexible tubes along the cables path inside the silicone. It allows the cables to slip with low friction. In the simulation, the additional rigidity created by these tubes are modeled using stiff springs. Two versions of the robots have been built through time (see Figures 3.17 and 3.19).

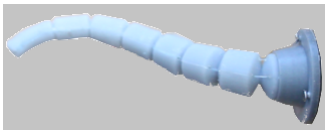

Figure 3.16: *Soft trunk #1.*

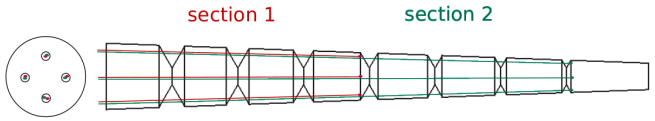


Figure 3.17: *Sketch of the soft trunk shown in Figure 3.22.*

Figure 3.18: *Soft trunk #2. Second design for grasping tasks.*

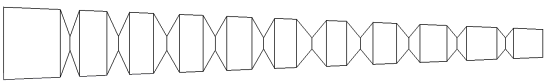

Figure 3.19: *Sketch of the soft trunk shown in Figure 3.18. The trunk is a bit longer and have more sections (each section being smaller) than the first design, given this new trunk more flexibility.*

The second version being an upgrade to make the trunk able to grasp objects.

What mainly changed is that the trunk is a bit longer, and have more sections (each section being smaller) than the first design, given this new trunk more flexibility. In the experimental scenario of Figures 3.20 and 3.22, we control only the tip of each trunk, even though more points could have been controlled. For instance, if we want to control the posture of the robot, we could add the control of the middle point of the trunk, allowing to reach S-shape configuration. In Figure 3.20, we show 2D trajectories of the trunk #1 end-effector for both real robot and simulated model. The average error is $4.7mm$. In Figure 3.21, we show 2D trajectories of the trunk #2 end-effector for both real robot and simulated model. The average error is $4.2mm$. Note that, we recently noticed that the cables we used in our experiments (fishing line) were not perfectly inextensible, which is not considered in our simulations. We could then, most probably, get better results with more appropriate cables. In Figure 3.22, we show a visualization of the trunk #2 end-effector control, with both real robot and simulated model. More experiments on these trunks are presented in chapter 4 and chapter 5.

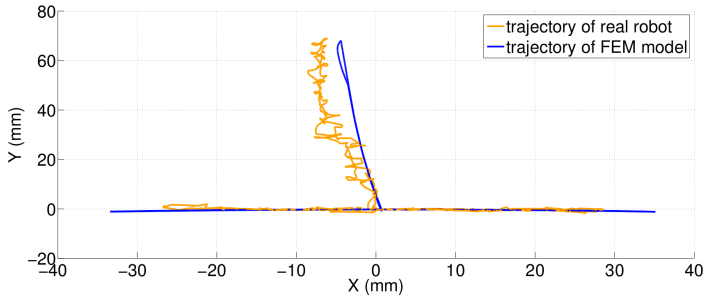


Figure 3.20: *2D trajectory of the trunk #1 end-effector.*

## 3.5 Applications

In this section we present two direct applications of our methods. The first one has been the subject of a published work in the medical community (Coevoet et al. 2014, 2015) for the clinical case of adaptive radiotherapy. It is not directly linked to soft robot as we did this work when our team was transitioning from medical simulation to soft robotics. In practice, the methodology is the same. The second application is an unpublished work, this time intended to graphics community, for the control of actuated puppets.

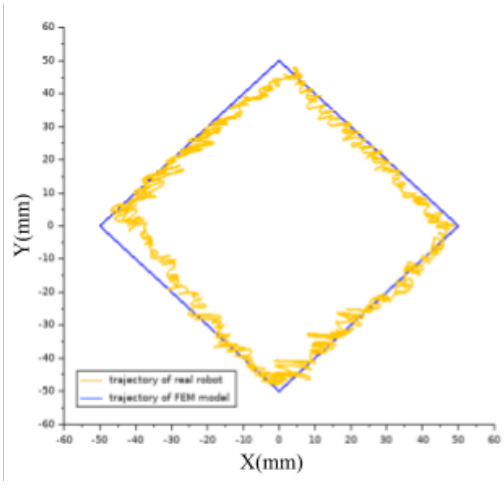


Figure 3.21: *2D trajectory of the trunk #2 end-effector.*

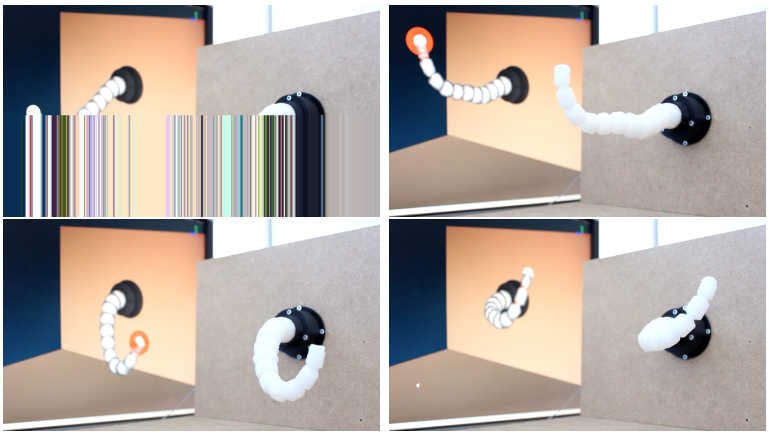

Figure 3.22: *The soft trunk #2 end-effector is controlled using our controller. Visual comparison between simulation and reality.*

### 3.5.1 Adaptive radiotherapy

In (Coevoet et al. 2014), we introduce a new methodology for semi-automatic registration of anatomical structure deformations. Given a set of few registered points provided by the user, the real-time optimization adapts the boundary conditions and(/or) some parameters of the FEM in order to obtain the adequate geometrical deformations. The approach is employed in the context of radiotherapy of the head and neck cancer. Radiotherapy treatment is established by using a treatment planning system (TPS), which combines patient medical images, radiation transport simulations and optimization algorithms in order to expose tumors to X-rays while sparing healthy structures. Yet, the treatment of the head and neck cancer can take seven weeks, and one observe weight loss on the patients due to the treatment. This loss of weight modifies the volume of the anatomical structures and induces large deformations, resulting to a TPS that may radiate healthy structures. In particular, sensitive structures such as the parotid glands may cross the target volume of radiation (see Figure 3.23), which leads to adverse effects for the patient. A registration of the structures on the TPS is then required.

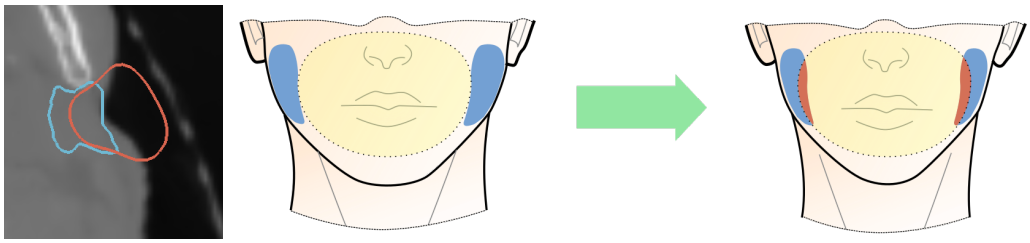

Figure 3.23: *Volume loss of parotids: (Left) segmentations of the parotids at weeks 1 (red) and 6 (blue). It is worth noticing the volume loss of the parotid as well as the motion of the center of mass. These two parameters have been used to characterize the deformation of parotids in (Barker et al. 2004). (Right) Due to weight loss, parotids may intersect the target volume (in yellow).*

#### Problem statement and motivations

The challenge remains on the registration method over the seven-week period. While significant work have been achieved in the field of automatic non-rigid registration (the reader may refer to (Crum et al. 2004) for a survey), these methods do not provide an easy control for the physicians. These algorithms also lack robustness and consistency when images are very complex, which is the case here: parotid glands are not easy to distinguish on images. On the contrary, dealing with manual segmentations and/or registrations is time-

consuming for the physicians and is not a viable solution for adapting the planning along with the treatment of the patients.

Consequently, the work proposed in (Coevoet et al. 2014, 2015) aim at providing a registration tool that can be driven by the physician (for robustness problem) with a very simple interface, and with an easy and explicit (*i.e.* no black-box tool) control on the parameters that have been used for the registration. And note that unlike other registration tools, our application do not rely on the image pixel grey-values. For this application, the registration of the parotid glands (or parotids) should be done using the parameters that are used in clinical studies: an observed volume loss and motion of the center of gravity (Barker et al. 2004).

### Registration tool

Our application starts with the geometrical models of the parotids that have been segmented during the initial planning and a CT image of the patient, after several weeks of therapy. First, an automatic rigid registration between the meshes and the new image is performed using the position of mandible bone. Then, the physician is asked to pick several points on the surface of the mesh and register them on the image (see Figure 3.24).

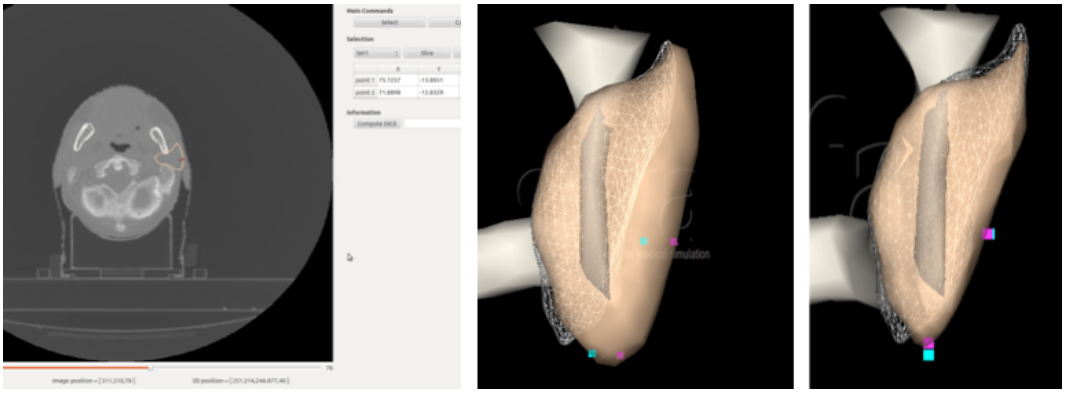

Figure 3.24: *Registration of the parotid deformations: (left) user interface that allows to select 2D points to be registered. (middle) in purple, points to be registered on the targeted points (blue). (right) parotid deformation after our inverse simulation. The corresponding manually segmented parotid is shown by the gray mesh.*

As this registration is done on a 2D slice of the 3D image, each registered point creates a 2D constraint. 3D registration is achieved when the user places points on different slices. The inverse simulation starts when the number of registered directions is superior to the number of unknowns.

Practically, a maximum of five values are retrieved during the optimization, so three registered points (since each register point induces two constraints) are sufficient. To improve the precision of the registration, more parameters or boundary condition values can be optimized, but the user will have to register more points.

### Validation study

A validation study conducted on seven patients exhibit results, whose quality is comparable with manual segmentation / registration while requiring significant less manpower. The decrease of patient exposure to radiations is also highlighted when using our results for adapting the TPS. We tested our approach on a *ground truth* set of seven patient datasets that contains the 3D images of the CT scan done every week of the therapy (total: 7 patients $\times$ 6 weeks = 42 CT scan).

Comparison between manual segmentations of the parotids and our method is achieved on all available CT scan by computing the DICE coefficient. A single dataset (six CT scan) has been manually segmented by two radiologists, and an average DICE coefficient of 0.7 has been computed to serve as a reference for the quality of our method. On these data, our method can be executed very quickly (completion of the registration is done in a single minute) with respect to a full manual segmentation making it compatible with the time constraints of a clinical routine. The graph in Figure 3.25 (left) illustrates that the parotids deformation is significant and second that our method exhibits good similarity compared to manual segmentation (average DICE between [0.8;0.9]).

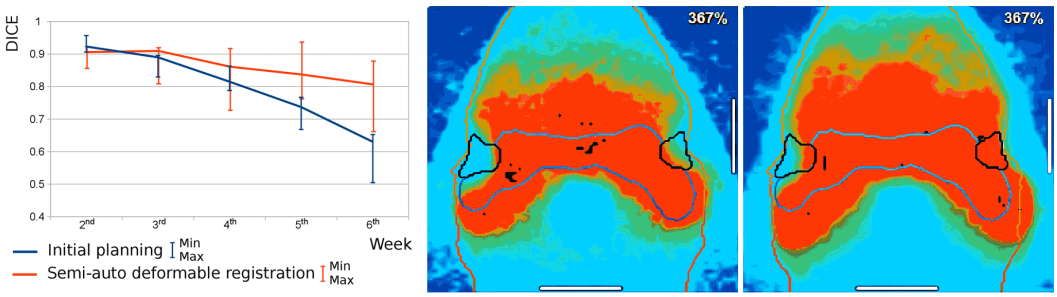


Figure 3.25: *Validation: (left) similarity between the initial segmentation and* ground truth *geometry in blue curve illustrates the deformation of the parotids (DICE decreasing), the red curve exhibits the good similarities between our semi-automatic registration and the* ground truth *geometry; (middle) planning adaptation using our registration vs no planning adaptation (right). The measured radiation is much lower when the planning is adapted.*

A dataset was selected for which the deformations were important and the parotids were not infiltrated by the tumor (therefore out of the target volume). We have a closer look at the last session of the therapy and particularly at the irradiation map of the parotids without considering planning adaptation Figure 3.25 (right) and with planning adaptation using our method to register the right parotid (middle). The resulting maps from the TPS show that the irradiation of the right parotid is significantly reduced and may limit the appearance of irradiation side-effects.

### 3.5.2 Soft robotics for entertainements and art

3D printing allows to quickly and easily transform a virtual 3D model into a real-world object. With this application, we want to follow the same approach by transferring a character animation to a tangible, deformable and robotized puppet.

#### Problem statement and motivations

The creation of new real objects is widely used in sculpture art, but also in stop-motion films. The use of tangible puppets has the potential of enhancing the aesthetics of the animation, at the price of a huge amount of manual work to create the animation. With animatronics, remotely controlled in live by technicians, the use of automation brings a greater realism and allows to pre-record some complex sequences of motions. If silicone or latex are often used for animatronic exterior envelope, their motion is induced by a rigid skeleton and a complex mechatronic system. In practice, they are often used for facial expression.

Soft, flexible and actuated objects have been recently used in an artistic context, to create tangible deformable objects (Bächer et al. 2012). Finding the suitable shape, material design or distribution, for reproducing a pose or a desired motion on a deformable figure requires new algorithms (Pérez et al. 2015), (Chen et al. 2014), (Schumacher et al. 2015), (Thomaszewski et al. 2014).

#### Experiments

We propose to use our algorithm to address the issue of controlling these soft actuated characters. We tested our algorithm on several built puppets. One of them is shown in Figure 3.26. We made this real puppet inspired by the *Marvel* character *Baby Groot*, from the movie *Guardians of the Galaxy*. We reproduced the dance sequence that this character performs in the movie, using the open source 3D software Blender. We give the animation as the input to

our simulation that outputs the cables force to animate the real prototype. The puppet is entirely made of silicone (Young's Modulus $\simeq 350kPa$) and is actuated like a classic string puppet, using 12 cables going through the hands and head. As in the movie, the pot of the puppet stays motionless on the ground. Note that the resulting motion of the puppet only matches the movie dance approximately. This is mainly due to the fact that we do not optimize the cables placement from the input animation. Yet the work space of the robot is limited by the cables placement.

Figure 3.26: *A puppet representing the character Baby Groot, actuated by 12 cables going through the hands and head and. Left: the inverse simulation. The white spheres represent the keypoints target. Right: the real prototype controlled by the simulation.*

On can also control puppets from motion capture. With the actuated soft octopus shown in Figure 3.27, we propose an interactive control of the puppet motion. The octopus is actuated with 12 strings : five of them are passing through each tentacle allowing to roll them up and out, five are attached to the middle up of each tentacle allowing to move them up and down and the final two are attached to each eye. An extra cable attached to the top of the puppet body allows to keep the octopus above the ground. This extra cable is not actuated. As mentioned in section 3.2.4, in the simulation, the cables are considered straight. To ensure consistency, the strings of the real puppet also have to be straight. As the arms of the character are thin pieces of silicone, and were not heavy enough to extend the cables, we had to weight each hand

of the puppet (in both reality and simulation).

We use a Leap Motion sensor to remotely control the motion of the Octopus in an interactive way. The sensor device tracks the fingers of the user and gives us each finger tip and intermediate positions. We map these positions on each tentacles, so that one finger controls one tentacle. With these two positions as input to our simulation, we are able to describe a tentacle rolling round and out, going up and down (see Figure 3.27).

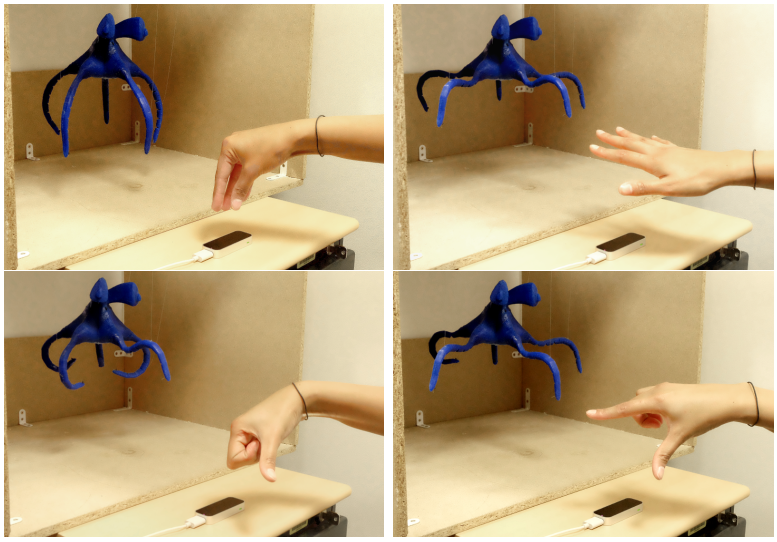

Figure 3.27: *An octopus soft robot made of silicone actuated with cables. Using a Leap Motion sensor, the fingers of a user are tracked. Their displacements are then used as an input for our simulation that outputs forces on the cables to deform the octopus according to the user motion.*

The two main computation steps of the simulation are the computation of the matrix $W$ and the resolution of the QP problem. In Table 3.6, we show the computation time[7] of our method. In particular, we show the average computation time for these two main steps, as well as for one step of the simulation (including graphics rendering).

Note that for the octopus example, we need to take into account the self-collision regions of each tentacle (at each *articulation*). This addition of contacts into the optimization process is not straight forward, specially when targeting real-time simulations. The method is presented in the next chapter 4.

[7]On a desktop computer with an i7 Intel processor 3.60GHz, and a Nvidia NVS510 with 192 Cuda cores

| Simulation | #DoFs | #Nodes | #Cont. | $W$ | $QP$ | Sim. |
|---|---|---|---|---|---|---|
| *Baby Groot* | 2034 | 678 | 0 | 27.9 $ms$ | 0.4 $ms$ | 60.2 $ms$ |
| Octopus | 2280 | 760 | 35 | 26.1 $ms$ | 16.4 $ms$ | 88.7 $ms$ |

Table 3.6: *Number of DoFs, number of nodes and the average number of contacts, computation time in ms of the matrices $W_{ij}$ construction, sequence of QPs resolution, and one time step of entire simulation.*

In the octopus example, where no part of the puppet is fixed, the structure shows a tendency to swing (as a global motion). We think this behavior is mainly due to the hardware (i.e. high variation of velocities in the servomotors). For such case, the current experiments lack a control strategy like proposed in (Santina et al. 2018) and (Thieffry et al. 2018) to deal with the dynamic effects.

## 3.6 Conclusion

In this chapter we have presented the mathematical basis of the proposed framework, that targets the design, simulation, and control of soft robots. This framework relies on a FEM approach to handle the mechanical deformations of the robots. Thanks to a set of Lagrange Multipliers defined on the boundary conditions, actuators and effectors are modeled accurately. These models allow for a forward kinematics and dynamics of the robots. Moreover, the mechanical representation can be used as an IP optimization that automatically computes the actuation to obtain control in the task space.

The capabilities of this framework are illustrated with several experiments showing that a reasonable accuracy between simulated and real soft robots can be obtained. Matching real-time performance was possible on both the forward and inverse problems by using relatively coarse FE meshes or the reduction method described in (Goury & Duriez 2018).

The modularity of the framework encourages many extensions. For instance, future work may include adding more complex mechanical laws, or adding robust control laws. This will increase the computational footprint of the simulation, whereas the short computation time needs to be maintained in order to do online control of the robot. Therefore, advanced numerical methods such as dedicated solvers should be considered to achieve sufficient accuracy without increasing the computation cost of the simulation.

An other extension, that is essential yet not straight forward, is the inclusion of contacts into the IP. In the next chapter we detail the method we propose

to control soft robots that interact with their environment, and in a interactive manner. That is, a controller with real-time performance allowing an online control of the robots in a unstructured and dynamic environment.

Chapter 4

# Inverse Model with Contact Hanlding

## Contents

## 4.1 Introduction

As soft robotic applications generally involve interaction of the robot with an environment, we propose, in this chapter, to extend the optimization problem detailed in section 3.3.2, to handle contacts.

Using a controller with no knowledge of the contacts that occur on the real robot may have two adverse effects that are pointed out in (Yip & Camarillo 2014), and illustrated in Figure 4.1. Since the model cannot sense the obstacle: first, an actuation may have an opposite effect on the effector position, that is opposite to what is modeled. More precisely, the line in the Jacobian matrix of the robot $W_{ea}W_{aa}^{-1}$, corresponding to the insertion actuation in Figure 4.1 (left and middle), is almost inverted. The second effect is artificial singularities as illustrate in Figure 4.1 (left and right), that is singularities in the model that do not occur on the real robot. These examples show the sake of adding contacts knowledge into the model, and since most contacts are unpredictable, we should be able to manage them online.

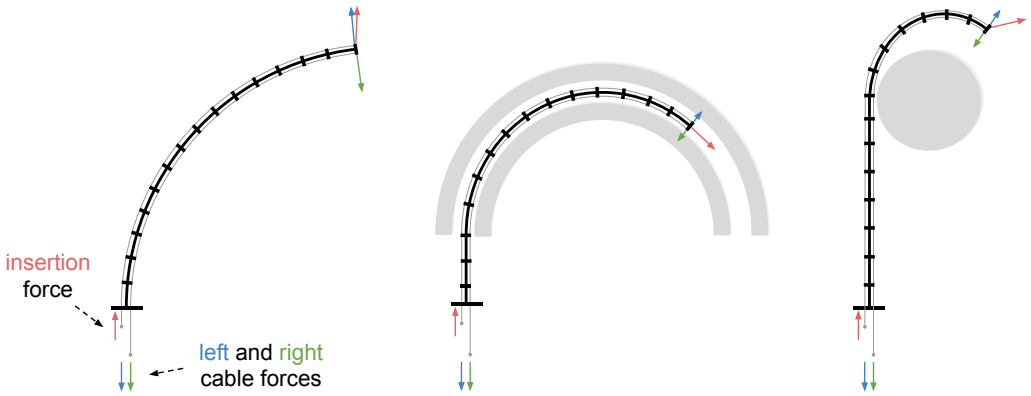


Figure 4.1: *Scheme of a soft rod actuated with two cables. Illustration of the effect of contacts on the model, highly inspired from the illustration given in (Yip & Camarillo 2014). Each rod is subject to the same cables displacement. (left) Rod without obstacle. (middle) The direction of the insertion actuation in inverted due to the obstacle. (right) No more singularity between insertion force and left cable actuations is experienced due to the obstacle.*

In the following, we detail the method we propose to include contacts into the optimization process. The second section 4.2 of this chapter present the techniques used for contact detection and modeling. We then detail the equations to solve in the third section 4.3. The main challenge being to maintain computation with real-time performance, we propose to use a specific solver

that we present in section 4.4. We finally give experimental results in section 4.5, direct applications in section 4.6, and conclude in section 4.7.

## 4.2 Contact model

To account for contact forces between the robot and its environment (and self-collisions) in the modeling, our simulation framework must allow for contact detection and response. Adding the management of contacts into the controller brings several computational problems.

First, we must locate the contact points. That is, we must use complex algorithms to detect when two objects collide, determine the points of contact and the direction of the response force. The task is more complicated for deformable objects that rigid ones (Teschner et al. 2005). For example, depending on the applications, it is often that rigid body collisions can be solved by only detecting one contact point, while with deformable object we may experience local deformations that require to consider multiple contact points. We present in this section the two detection methods we used for this work. This detection step is important as each contact point introduces a new variable into the optimization problem (more when considering friction), and the system can become huge and expensive to compute.

Second, for contacts to be realistic we must follow physics law. We will use complementarity conditions, in accordance with Signorini's law, for frictionless contact. This complementarity condition complicates significantly the equations to solve. As for actuation, we use a constraint approach for contact response. We briefly talk about other possibilities and present how the equations of motion are updated in consequence.

### 4.2.1 Detection

The exact moment of collision between two bodies is difficult to determine. That is why we use the time stepping scheme described in 3.3.1. During the free motion, objects that intersect each other must be identified. This can be done in our framework by using a *bounding volume hierarchies* approach coupled with a *local minimum distance* method, or a private implementation of the GPU approach proposed in (Allard et al. 2010) based on *layer depth images*. Other collision detection techniques, such as distance fields and spatial partitioning are discussed in (Teschner et al. 2005).

### Bounding volume hierarchies and local minimum distance

The idea of the bounding volume hierarchies approach is to enclose subsets of the object's primitives (i.e. edges, triangles, polygons) with bounding volumes (e.g. box, sphere). For example, one can subdivide volumes from top to bottom, and obtain a tree with the root being the volume that contains all the primitive (see Figure 4.2). Thus, is each node in the tree is associated with a subset of the primitives of the object, together with a bounding volume that encloses this subset with a smallest containing instance of some specified class of shapes. Here we use boxes.

During a first phase, pairs of subsets that are not colliding are eliminated by estimating the distance between the bounding volumes. This step is executed in a hierarchical manner, that by exploring the tree from top to bottom. When primitives that are closed to collide are identified, we now have to use an other method to determine exactly which primitives are in contact, and what are the interpenetration distances.

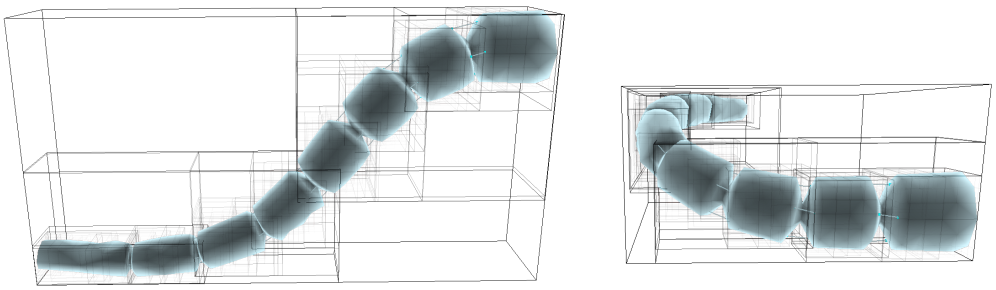

Figure 4.2: *Illustration of the bounding volume hierarchies trees for the trunk. As the structure moves and deforms the bounding trees have to be updated. In comparison with rigid objects, with deformable objects these computations are done more frequently.*

One method we often use to determine the contact points is based on minimal distances computation using an implementation of the algorithm described in (Johnson & Willemsen 2003), adapted to deformable meshes. This algorithm easily manages contact detection between concave meshes, while limiting the number of couples of proximity points, as it selects a couple of points only if they represent a local minimum distance. We can also use an adaptation of the algorithm for self collision, but in practice, we can often predefine the two points of the mesh that will self-collide, and simplify the self-collision detection.

**Volume constraints using layer depth images**

In (Allard et al. 2010), Allard et al. propose a method based on layer depth images and volume constraints. Layer depth images structure is a stack of 2D images representing the object surfaces, and are used here to determine the interpenetrations between objects (see Figure 4.3).

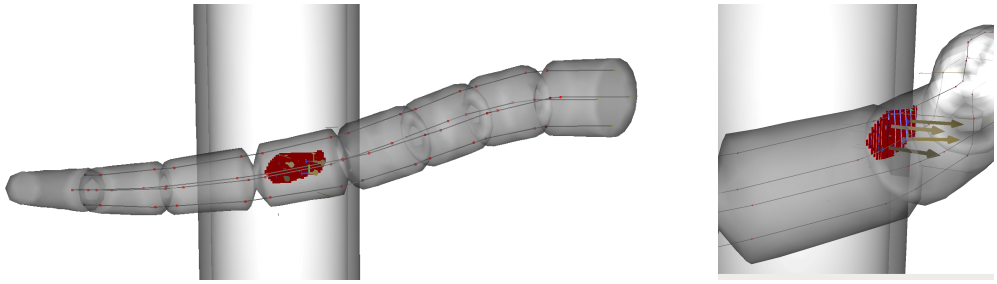

Figure 4.3: *Illustration of the volume constraints method proposed in (Allard et al. 2010) using layer depth images. Intersection volumes.*

In each pixel of the images is stored the surface depth in the viewing directions (x,y, and z), as well as additional data such as normal orientation or object index. By sorting the depths stored in the different images at each pixel, one obtain the ordered list of surface intersections with a ray parallel to the viewing directions. The method has the advantage of providing smaller systems of equations than other traditional mesh contact models, when applied to complex geometries. The computation of the layer depth images is done fast thanks to the use of GPU.

### 4.2.2 Signorini's law

To obtain a realistic behavior in simulation when dealing with contacts, we must follow physics law. For frictionless contact, we follow Signorini's law (Kikuchi & Oden 1988), which is the simplest law to model contact. First, this law guaranties no interpenetration between objects, by setting that the distance between two objects must be positive or zero. Second, it ensures that the contact forces are well oriented, by forbidding attraction force. Finally only one of these two quantities can be non zero at a time. Either (1) the contact is active; the distance between the two objects at the contact point is equal to zero and the force can be non zero, or (2) the contact is inactive; the force at the detected potential contact point must be zero and the distance is non zero.

This is formalized by the following complementarity condition:

$$0 \leq \lambda_c \perp \delta_c \geq 0 \tag{4.1}$$

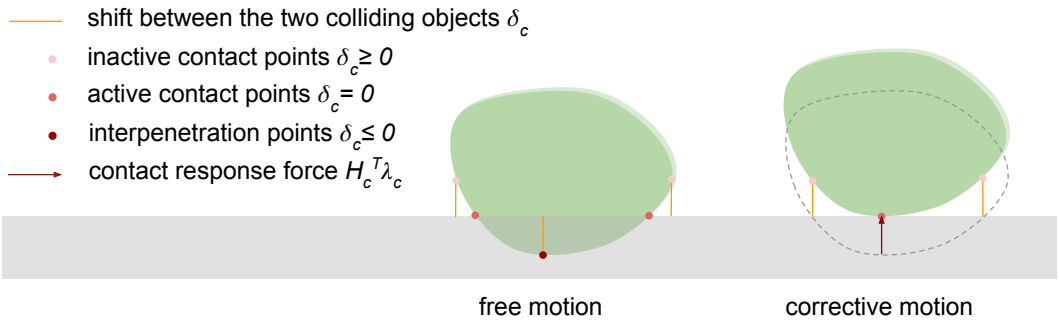


Figure 4.4: *Illustration of Signorini's law and quantities of equation* (4.1).

where $\delta_c$ is the shift between the two objects, and $\lambda_c$ is the intensity of the contact force. Using the geometric mapping function between the contact distance $\delta_c$ and the position of the DoFs $x$, we can build a Jacobian of contact $H_c(x)$. The direction of the force (normal to the surface) is thus held in the matrix $H_c$, so that the product $H_c^T \lambda_c$ corresponds to the contact force. See Figure 4.4 for an illustration.

### 4.2.3 Constraint response

In computer graphic community, contact response are often computed using penalty-based method. Penalty-based contact model have several drawbacks. The idea is to create a spring between the two object at the contact point (usually an implicit spring model is used for stability purposes), which give a coupled system of the two colliding objects. Thus, this method involves to solve large systems of equations. Furthermore, this technique is not physically correct, and the results highly depend on the stiffness chosen for the springs. It may result to interpenetration of the objects (if the stiffness is too low), or hard take off (if the stiffness is too high).

For this thesis we use a constraint response approach. Contacts are then adding to the system using Lagrange's multipliers, as for actuations, yielding to the following problem:

$$\begin{cases} Ax = b + H_a^T \lambda_a + H_c^T \lambda_c \\ 0 \leq \lambda_c \perp \delta_c \geq 0 \end{cases} \tag{4.2}$$

Note that, by using Signorini's condition in this chapter, we do not yet consider friction effects.

## 4.3 Solving the constraints

We have now, two additional unknowns $\delta_c$ and $\lambda_c$ that are considered in the system, and also linked by equation (3.13). In addition, these two values are also linked by the dynamics. In the case of multi-contact, any contact force can modify the distance between the couple of any contact points. Let us just rewrite equation (3.13) by including constraints on actuators and Signorini's complementarity condition, yielding to:

$$\begin{bmatrix} \delta_e \\ \delta_a \\ \delta_c \end{bmatrix} = \begin{bmatrix} W_{ea} & W_{ec} \\ W_{aa} & W_{ac} \\ W_{ca} & W_{cc} \end{bmatrix} \begin{bmatrix} \lambda_a \\ \lambda_c \end{bmatrix} + \begin{bmatrix} \delta_e^{\text{free}} \\ \delta_a^{\text{free}} \\ \delta_c^{\text{free}} \end{bmatrix} \tag{4.3}$$

$$\delta_{max} \geq \delta_a \geq \delta_{min} \tag{4.4}$$

$$\lambda_{max} \geq \lambda_a \geq \lambda_{min} \tag{4.5}$$

$$0 \leq \lambda_c \perp \delta_c \geq 0 \tag{4.6}$$

where matrices $W_{ij} = H_i K^{-1} H_j^T$ $(i, j = e, a, c)$ are homogeneous to a compliance, and now gather the mechanical coupling between effector points $e$, actuators $a$ and contacts $c$. The quantities of the above equations are reminded in Figure 4.5. We note respectively $n_a$ and $n_c$, the number of actuators and the number of contact forces.

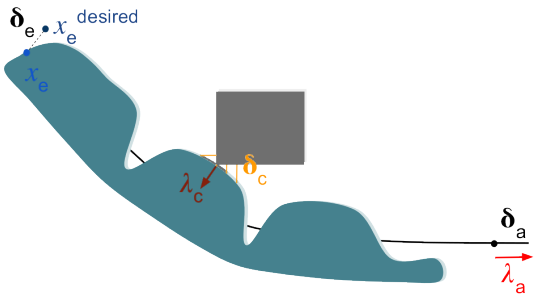


Figure 4.5: *Illustration of the quantities of equation (4.3) for a deformable robot with cable actuation (in blue), in the presence of an obstacle (grey square).* $x_e^{desired}$ *is the desired position for the controlled point* $x_e$.

### 4.3.1 Forward problem

For the forward case, the actuations are known, but we still have to solve the contacts. In case of contacts only, the problem takes the form of the following Linear Complementarity Problem (LCP):

$$\begin{cases} \delta_c = W_{cc}\lambda_c + \delta_c^{free} \\ 0 \leq \lambda_c \perp \delta_c \geq 0 \end{cases} \tag{4.7}$$

If we add some equality constraints to apply the motion created by actuators (we can either set $\lambda_a$ or $\delta_a$), we obtain the following Mixed Complementarity Problem (MCP):

$$\begin{cases} \delta_c = W_{ca}\lambda_a + W_{cc}\lambda_c + \delta_c^{free} \\ 0 \leq \lambda_c \perp \delta_c \geq 0 \\ \delta_a = l \end{cases} \tag{4.8}$$

In both cases, we use a block Gauss-Seidel like solver to find a solution (see (Duriez et al. 2006) for details). Then, this solver provides the values of $\lambda_c$, and is followed by the computation of the final configuration. That is by extended equation (3.14) to:

$$x = x^{free} + A^{-1}H_a^T\lambda_a + A^{-1}H_c^T\lambda_c \tag{4.9}$$

### 4.3.2 Inverse problem

For the inverse case, we want to optimize the intensity of efforts $\lambda$ so that effectors reach desired positions, while satisfying Signorini's law for contacts. The inclusion of contacts into the problem, does not allow to obtain a condensed relation between displacement of effector given the displacements of the actuators, as given in equation (3.15).

To find the efforts $\lambda_a$ and $\lambda_c$, we minimize the norm $\|\delta_e\|$, subject to the constraints on actuators course (4.4), the constraints on actuators efforts (4.5), and Signorini's condition (4.6), yielding to the following Quadratic Program with (linear) Complementarity Constraints (QPCC):

$$\min_{\lambda_a,\lambda_c} \left\| W_{ea}\lambda_a + W_{ec}\lambda_c + \delta_e^{\text{free}} \right\|^2 \tag{4.10}$$

$$\text{s.t. } C \begin{bmatrix} \lambda_a \\ \lambda_c \end{bmatrix} \geq d \tag{4.11}$$

$$0 \leq \lambda_c \perp \delta_c = W_{ca}\lambda_a + W_{cc}\lambda_c + \delta_c^{\text{free}} \geq 0$$

with equation (4.11) being a rewriting of constraints on actuation (4.4) (i.e. limit of cable displacement or volume growth for instance). Note that $\lambda_c$ is part of the optimization variables, which allows the controller to make use of contact forces to achieve the desired motion. Solving this problem in real-time is challenging due to the complementarity constraint. In the following we detail the specific solver we propose.

## 4.4 QPCC solver

Recent work in optimization have addressed the problem of linear and quadratic programs with linear complementarity constraints (Hu et al. 2012, Bai et al. 2013). They seek to find the global optimum of the problem and demonstrate that it can be accomplished in finite time. However, as finding the global minimum is difficult to achieve in real-time, we developed our own specific solver based on the decomposition method as mentioned in (Chen & Goldfarb 2007).

### 4.4.1 Decomposition method

The complementarity constraints (4.1) defines $2^{n_c}$ choices. Each of them can be characterized by a subset I of $\{1, \ldots, n_c\}$ giving the elements of $\lambda_c$ that are forced to be zero. Let $e_i$ be the $i$-th column of the $n_c$-by-$n_c$ identity matrix, and define $S_{\text{I}}$ as the matrix whose columns are the $e_i$ for $i$ in I. Given a matrix $M$, $MS_{\text{I}}$ (respectively $S_{\text{I}}^T M$) selects the columns (respectively rows) of $M$ indexed by I. Likewise, we define $\bar{S}_{\text{I}}$ for $i$ not in I. $\begin{bmatrix} S_{\text{I}} & \bar{S}_{\text{I}} \end{bmatrix}$ is a permutation (and thus orthonormal) matrix. Given a complementarity choice I, QPCC (4.10) rewrites:

$$\min_{\lambda_a,\lambda_c} \left\| W_{ea}\lambda_a + W_{ec}\lambda_c + \delta_e^{free} \right\|^2 \tag{4.12}$$

$$\text{s.t. } C \begin{bmatrix} \lambda_a \\ \lambda_c \end{bmatrix} \geq d \tag{4.13}$$

$$S_{\mathrm{I}}^T \lambda_c = 0 \tag{4.14}$$

$$S_{\mathrm{I}}^T (W_{cc}\lambda_c + W_{ca}\lambda_a + \delta_c^{\mathrm{free}}) \geq 0 \tag{4.15}$$

$$\bar{S}_{\mathrm{I}}^{\ T} \lambda_c \geq 0 \tag{4.16}$$

$$\bar{S}_{\mathrm{I}}^{\ T} (W_{cc}\lambda_c + W_{ca}\lambda_a + \delta_c^{\mathrm{free}}) = 0 \tag{4.17}$$

This is a QP piece of (4.10) we refer to as $\mathrm{QP_I}$. We propose an iterative method that starts from an initial feasible set I. After solving $\mathrm{QP_I}$, we inspect the state of the inequality constraints. If an inequality constraint has reached its boundary at the end of the optimization, it means that the solution may potentially be further optimized by pivoting the corresponding constraint. We set each inequality constraint that reached their boundary at the end of the optimization as candidate for pivot. In our algorithm we pivot one constraint at a time.

As mentioned in (Chen & Goldfarb 2007), one way to determine which constraint should be the best candidate for pivot, is to examine the values of the dual variables. These variables are provided to us by the solver (of the qpOASES library) that we use to solve each $\mathrm{QP_I}$, and come from the Lagrangian dual problem of the $\mathrm{QP_I}s$ (see e.g. (Ferreau et al. 2014) for more detailed explanations). In our implementation, we select for pivot, the candidate with the greater dual variable.

By pivoting the corresponding complementarity constraint we get a new QP with a different set of linear constraints. We solve this new problem and repeat the process until there is no more candidate for pivot, effectively solving a sequence of $\mathrm{QP_I}$. A proof of the convergence of this kind of algorithm to a stationary point is given in (Giallombardo & Ralph 2008).

Note that, as the solver converges to a stationary point and not the global solution, the contacts have to be active ($\lambda_c > 0$) at the beginning of a QP resolution to be exploited by the actuation.

### Initialization

As mentioned above, our solver requires an initial feasible set I. In the implementation, this initial set is found by solving the contacts as a Linear

Complementarity Problem (LCP) while considering the actuator force $\lambda_a$ constant:

$$\begin{aligned} \delta_c &= W_{cc}\lambda_c + W_{ca}\lambda_a + \delta_c^{free} \\ 0 \leq \lambda_c &\perp \delta_c \geq 0 \end{aligned} \tag{4.18}$$

This system has a solution for any $\lambda_a$ because $W_{cc}$ is positive definite (Murty 1972) and this solution is unique. Thus there is always at least one feasible set I. The initial guess of $\lambda_a$ is either 0 or the solution of the previous QPCC optimization when it is available (warm start).

### 4.4.2 Reduced formulation

The above scheme can be improved by taking into account the specificity of our problem. Indeed, $W_{cc}$ is positive definite (because $A$ is positive definite). As a result, $\lambda_c$ is an affine function of $\lambda_a$ for a given I. This allows to remove $\lambda_c$ from $\mathrm{QP_I}$ and solve a smaller problem. We name this the *reduced formulation*. In the remainder of this section, we drop the index I for matrices $S_\mathrm{I}$ and $\bar{S}_\mathrm{I}$.

Using that $\lambda_c = \begin{bmatrix} S & \bar{S} \end{bmatrix} \begin{bmatrix} S & \bar{S} \end{bmatrix}^T \lambda_c = SS^T\lambda_c + \bar{S}\bar{S}^T\lambda_c$, we get from equation (4.14) that $\lambda_c = \bar{S}\bar{S}^T\lambda_c$. Reintroducing this result in equation (4.17), and solving for $\bar{S}^T\lambda_c$ yields:

$$\bar{S}^T\lambda_c = -\left(\bar{S}^T W_{cc}\bar{S}\right)^{-1}\bar{S}^T\left(W_{ca}\lambda_a + \delta_c^{\mathrm{free}}\right)$$

We define $M_I$ and $q_I$ such that the above relation rewrites $\bar{S}^T\lambda_c = M_I\lambda_a + q_I$. Then we have the affine relation

$$\lambda_c = \bar{S}\left(M_I\lambda_a + q_I\right) \tag{4.19}$$

and $QP_\mathrm{I}$ is equivalent to the following QP we name $QP_\mathrm{I}^r$:

$$\begin{aligned} \min_{\lambda_a}.\ & \left\| \left(W_{ec}\bar{S}M_I + W_{ea}\right)\lambda_a + W_{ec}\bar{S}q_I + \delta_e^{free} \right\|^2 \\ \text{s.t. } & C^r\lambda_a \geq d^r \\ & M_I\lambda_a + q_I \geq 0 \\ & S^T\left(W_{cc}\bar{S}M_I + W_{ca}\right)\lambda_a + S^T\left(W_{cc}\bar{S}q_I + \delta_e^{\mathrm{free}}\right) \geq 0 \end{aligned} \tag{4.20}$$

We obtain $\mathrm{QP_I^r}$ by solving equations (4.14) and (4.17), what would have been done anyway by the QP solver. But we did so by leveraging the structure of

the problem (with respect to $\lambda_a$ and $\lambda_c$), and we can take into account the fact that $\bar{S}^T W_{cc} \bar{S}$ is positive definite[1]. In particular, we can use the Cholesky decomposition of $\bar{S}^T W_{cc} \bar{S}$ to compute its inverse[2], which is much cheaper computationally than the more general decomposition the QP solver would need to use. As a result, given the matrices $W_{ij}$ and C, and the vectors $\delta_i^{free}$ and $d$, solving $\mathrm{QP_I}$ is slower than computing the matrices and vectors in $\mathrm{QP_I^r}$ and solving $\mathrm{QP_I^r}$.

The ratio between the two computation times depends on $n_a$ and $n_c$. When $n_a >> n_c$, the timings are the similar (in particular, if $n_c = 0$, both problems are the same). However, for a fixed $n_a$, the reduced formulation becomes better and better as $n_c$ increases. For example, for $(n_a, n_c) = (3, 3)$, the average ratio is 1.25. It is 1.88 for $(3, 10)$, and 7.82 for $(3, 50)$. Usually, $n_c$ is much larger than $n_a$ so that the reduced formulation has a real computational advantage. The computation time could be further reduced when we consider the sequence of resolutions performed in the iterative method described above. Indeed, in this case, two successive set I differ by only one element, so that we pass from one matrix $\bar{S}^T W_{cc} \bar{S}$ to an other by adding or removing one row and one column. Therefore, there is no need to compute the Cholesky decomposition from scratch at each iteration, but it can rather be updated. The full decomposition is in $O(k^3)$ where $k$ is the size of the matrix, while the update is in $O(k^2)$ (Golub & Van Loan 1996). Preliminary tests show that this reduces the computation time by $20 - 25\%$ for all but the first iteration. However, note that we do not use this improvement yet in the controller.

### Visualization

Another benefit of the reduced formulation, is that it allows to visualize the problem when $n_a$ is small (1,2 and partially for 3), even for complex contact configurations (i.e. with large $n_c$). This visualization is helpful to better understand the properties of the problem.

For a given I, the two last constraints of $\mathrm{QP_I^r}$ defines a polytope $P_\mathrm{I}$ (possibly empty or unbounded) in the space of $\lambda_a$. A face of this polytope corresponds to one line of these constraints holding as an equality, i.e. a change of complementarity choice. Since for every $\lambda_a$ there is at least one I (and that when there are more than one, this corresponds precisely to changes of complementarity

[1] It is a principal minor of the positive definite matrix $W_{cc}$.

[2] Note that, following the recommended practice, we do not compute explicitly the inverse. Rather (see Golub & Van Loan (1996)), given the Cholesky decomposition $\bar{S}^T W_{cc} \bar{S} = LL^T$, with $L$ a lower triangular matrix, we compute $\begin{bmatrix} A_\mathrm{I} & b_\mathrm{I} \end{bmatrix} = -L^{-T} L^{-1} \bar{S}^T \begin{bmatrix} W_{ca} & \delta_a^{\mathrm{free}} \end{bmatrix}$, where the left multiplication by $L^{-1}$ and $L^{-T}$ are obtained by forward and backward substitution.

choices), the union of these polytopes covers the whole space of $\lambda_a$. Two polytopes $P_{\mathrm{I}_1}$ and $P_{\mathrm{I}_2}$ may share a part of their boundaries, when one goes from $\mathrm{I}_1$ to $\mathrm{I}_2$ by at most $n_a$ changes (pivots).

Since for each $\lambda_a$ there is a unique $\lambda_c$, we can express the cost function of the QPCC (4.10) as a function of $\lambda_a$ only (for a given I, it is the cost function of $\mathrm{QP}_\mathrm{I}$). We denote as $c(\lambda_a)$ this function. It is defined by pieces, each piece support being a polytope $P_\mathrm{I}$. With this representation, the original QPCC can be seen as an optimization problem with a piecewise-defined cost function with only the constraint on the actuation:

$$\begin{aligned} &\min_{\lambda_a} \; c(\lambda_a) \\ &\text{s.t. } C\lambda_a \geq d \end{aligned} \tag{4.21}$$

When $n_a$ is 1 or 2, it is possible to draw the graph of $c$. We give examples of such graphs for $n_a = 2$ in Figures 4.6 and 4.8.

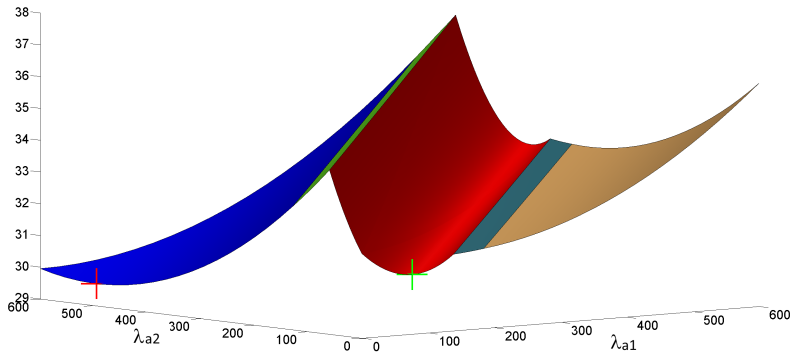


Figure 4.6: *Piecewise cost function for the problem corresponding to Figure 4.7 (left), with* $\lambda_a \in [0, 500]^2$. *Each colored region corresponds to a complementarity choice* I *and is supported by the corresponding polytope (polygon in the 2d case)* $P_\mathrm{I}$. *There are two local minima corresponding to pulling one cable or the other. Pulling slightly the upper cable (up to* $\lambda_{a1} \approx 75N$, *local minimum with the green cross) uppers the beam, but pulling more makes the end effector go down due to the upper contacts. Pulling on the lower cable makes use of the lower contacts to upper the end effector and reach the global minimum (red cross,* $\lambda_{a2} \approx 495N$*). One can see valleys along* $\lambda_{a1} = \lambda_{a2}$. *This is because the two cables are opposite, thus for a given* $\lambda_a$, $\lambda_a + (\mu, \mu)^T$ $(\mu \in \mathbb{R})$ *gives a similar (but not identical) end-effector displacement.*

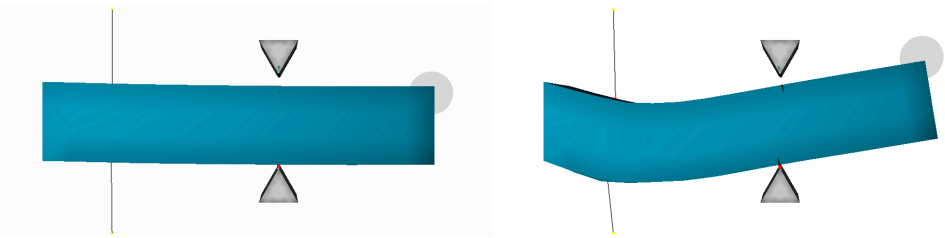

Figure 4.7: *Soft beam actuated with two cables. We control the beam end-effector position (target is the light grey sphere). The grey objects are fixed rigid obstacles. Possible contact points with each obstacle are detected, and contact with one obstacle is being reached and used to better solve the target.*

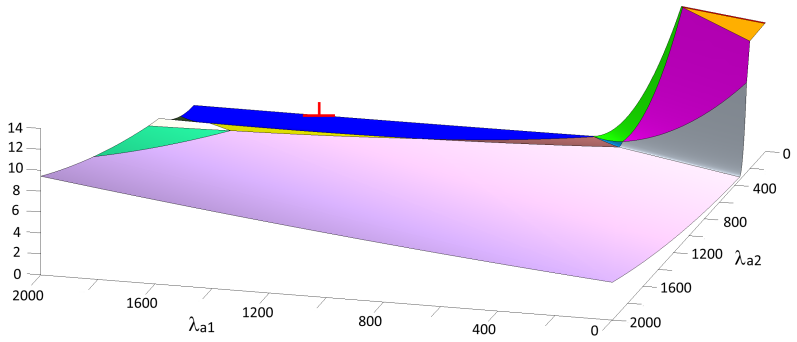


Figure 4.8: *An example for the same robot and different positions of obstacles, with* $\lambda_a \in [0, 2000]^2$. *Starting from* $\lambda_a = (0, 0)$ *(thin red zone in the upper right), our solver iterates across the regions in that order: red, orange, dark purple, green, cyan, brown, and finally blue, where the global minimum (red cross) is obtained. Note that the order to choose the pivot has an impact here: one could also have gone from the dark purple region to the gray one, and ended up in local minimum.*

## 4.5 Experiments and results

We now give results obtained from experiments. In the first section we discuss and compare results from the two different collision detection methods we can use. In the second section we describe the results of our controller with contact handling. First, on several numerical examples, and then on our soft trunk when colliding with fixed obstacles, along with obstacles that are tracked and moved dynamically. Performances are also given at the end of this second section. Note that, the deformable model we used for all the examples here is the co-rotational model.

### 4.5.1 Contact detection methods

The QPCC solver we propose to use provide fast computation time. However, if the number of contacts is too high, the computations will slow down. When the surfaces in contact is large due to local deformations for example, local minimum distance method may outputs a large number of contact points to solve the interpenetration (see Figure 4.9). In such cases, using the method based on layer depth images may provide the same results with less contact points, and then less computation time for the QPCC to solve the IP.

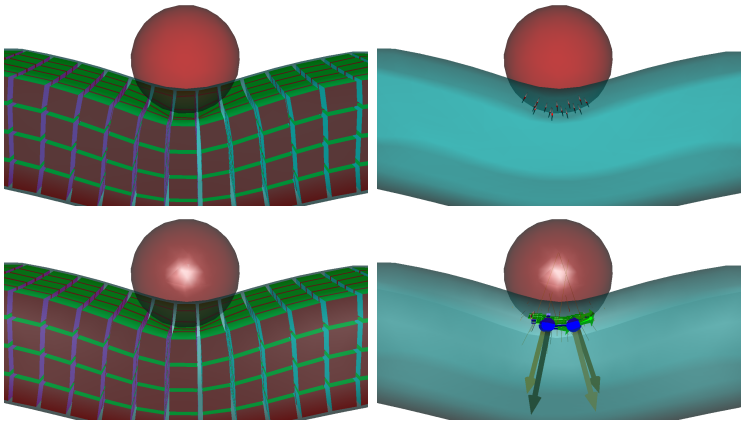

Figure 4.9: *(top) Bounding volume hierarchies and local minimum distance method. The method is using 15 contact points to solve the collision. (bottom) Volume constraints using layer depth images. With this method (with some parameters being set) only four contact points are used to solve the collision and obtain same deformation. (left) FE mesh of the soft beam is shown (hexahedras) to visualize the deformation. (right) Contact points computed by each method are shown.*

Depending on the robots' geometry, deformations due to contact may not be localized on the surface in contact, but somewhere else. For instance, in the trunk example (see Figure 4.3), the deformations occur mainly on the softer parts, that is where the section is reduced, and not so much on the contact surface, that is on each *segment* of the trunk (see Figure 4.10). Each *section* shows generally only displacement motions. In such cases, only few contact points are needed to solve the interpenetration, and using local minimum distance method will not increase the size, nor the computation time of the QPCC.

However, if it is needed to use a surface mesh with high resolution, the latter method will take time to compute the detection phase (see Table 4.1). In that case, method based on layer depth images (Allard et al. 2010) provide better

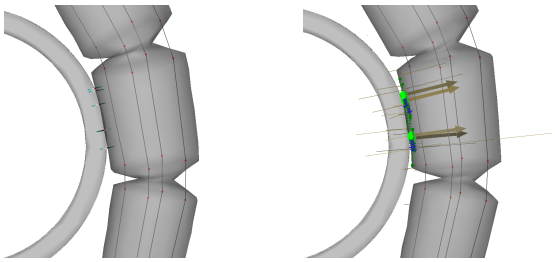

Figure 4.10: *(left) Bounding volume hierarchies and local minimum distance method. The method is using five contact points to solve the collision. (right) Volume constraints using layer depth images. The method uses four contact points to solve the collision and obtain same deformation.*

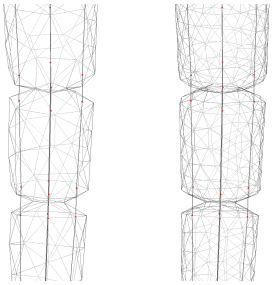

Figure 4.11: *(left) #1 Coarse surface mesh. (right) #2 Fine surface mesh.*

| | | BVH & LMD | | LDI | |
|---|---|---|---|---|---|
| Example | #Nodes | #Cont. | Time | #Cont. | Time |
| Soft beam | 434 (beam), 218 (sphere) | 15 | 1.25$ms$ | 4 | 1.76$ms$ |
| Trunk #1 | 338 (trunk), 273 (cylinder) | 7 | 5.80$ms$ | 4 | 2.56$ms$ |
| Trunk #2 | 1046 (trunk), 273 (cylinder) | 5 | 38.93$ms$ | 4 | 2.93$ms$ |

Table 4.1: *Mean computation time of the collision detection phase with respect to the method used, for the two examples given in this section. BVH & LMD: Bounding volume hierarchies and local minimum distance method. LDI: Volume constraints using layer depth images. #Cont.: number of contact points. #Nodes: number of nodes of the surface meshes (triangles) used for collision detection.*

performance results. Note that, with this GPU based method, parameters can be tuned to obtain more or less contact points. For example, in the trunk simulation, by divided by four the resolution (pixel size) of the images, we obtain 31 contact points in computation time of the same order (4.84$ms$).

### 4.5.2 Motion control

In this section we give results of our controller on both numerical and real experiments. As preliminary results (similarly to what we have done in chapter 3), for the numerical experiments, IK is confronted to the forward resolution. Then, results from real experiments on the soft trunk are presented, with both fixed and moving obstacles. We also provide visual results of self-collision

handling on a soft tentacle.

### Numerical experiments

We present here several numerical experiments showing the efficiency of the inverse algorithm with contact handling, in different scenarios. Each scenario shows a particular interest of the method: first, to solve self-collision regions, second, to solve collision with the environment.

**Self collision regions.** It is frequent that soft robots are designed with self collision regions. It may be mandatory to model these contacts in order to control the robot. For example, the actuation of the soft finger in Figure 3.8, and each tentacle of the octopus in Figure 3.27 required to take these self collision region into account, when displacement of the cables reaches a certain threshold.

Here, we give an other example where we simulate a soft body with four cavities actuated with pressure (see Figure 4.12). Each inflatable section being separated from the others by a self-collision region. In this particular example, having the controlled point (top of the body) reaches its desired position entail the use of self-collisions.


Figure 4.12: *Soft tower with four cavities actuated with pressure. The algorithm determines how to inflate each cavity to get the end-effector reach the desired position (white sphere). Self-collision points are represented by red lines.*

In the same way we have done in chapter 3, we confronted the results from the inverse resolution to those of the forward. We applied four different pressures in each cavity, and extract the position of the tip and middle points of the object (see Figure 4.13). We set these two positions as target in the inverse resolution. Note that, by controlling two positions, we guarantee that the problem is well-posed (i.e. that there is only one solution). The quantitative results are given in Table 4.2. The inverse resolution retrieves the pressure values with an error below 1% for each cavity.

Figure 4.13: *Pressure estimation. Two positions on the soft object are controlled, the tip and the middle.*

| Cavity | Applied pressure | Estimated pressure |
|---|---|---|
| #1 | $20kPa$ | $19.9kPa$ |
| #2 | $18kPa$ | $18.0kPa$ |
| #3 | $24kPa$ | $24.0kPa$ |
| #4 | $22kPa$ | $21.9kPa$ |

Table 4.2: *Performance of our approach to retrieve pressures applied in cavities of the model of Figure 4.13. The error for each cavity is below* 1%.

**Collision with obstacle.** This second and last numerical example is confronting again the inverse resolution to the forward one. Here, a soft finger actuated with two parallel cables is colliding with a fixed obstacle through its course (see Figure 4.14). In the forward resolution, we apply a displacement of cables ($18mm$ each), making the finger collide with the obstacle. We extract the resulting position of the end-effector. When setting this position as a target to the inverse resolution, the algorithm solve the end-effector position and with an error on the cable displacement below 1% (see Table 4.3).

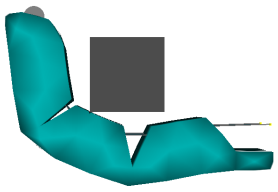

Figure 4.14: *Cables displacement estimation. We control the finger tip position.*

| Cable | Applied disp. | Estimated disp. |
|---|---|---|
| #1 | $18mm$ | $17.9mm$ |
| #2 | $18mm$ | $17.9mm$ |

Table 4.3: *Approach performance to retrieve displacement applied on the two cables of the finger in Figure 4.14. The error in cables displacement is below* 1%.

### Real experiments

We built few soft robots to tests the controller. Several results are presented in this section. First, we show results on self collision handling on one of the octopus tentacle, and second, we show results obtained on our soft trunk when colliding with fixed obstacles, and also when colliding with evolving obstacles that are tracked and moved by the user dynamically. In each case we control one or two points of the model and the target is interactively moved by a user

or follows a predefined trajectory. Note that, self collisions are also solved in the trunk example.

**Self collision regions.** We tested our modeling approach on the simulation of one tentacle of the octopus presented in section 3.5.2. The tentacle is actuated with one cable passing through its entire structure. By pulling the cable, we create contacts between the two sides of the articulations.

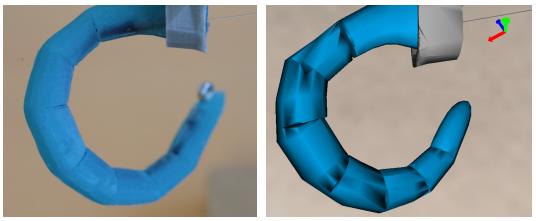

Figure 4.15: *Tentacle end-effector controlled using our algorithm for IK with self-collision regions handling. Left: real prototype. Right: simulated model.*

In Figure 4.15, we show that our simulation provides very similar configurations to those of the real prototype. Note that not taking these self collision region into account in the controller may provide very bad results.

**Soft trunk progression in unstructured environment.** We applied our method on the real cable-driven soft trunk-like robot described in section 3.4.3. In the experimental scenario shown in Figure 4.16, we control the tip of the trunk and the target follows a predefined trajectory. The real trunk is attached to a platform moving along the robot direction, allowing a forward and backward displacement. This actuation is also modeled in the simulation. Using the optimization, we were able to interactively drive the trunk end-effector between the two fixed rigid cylinders.

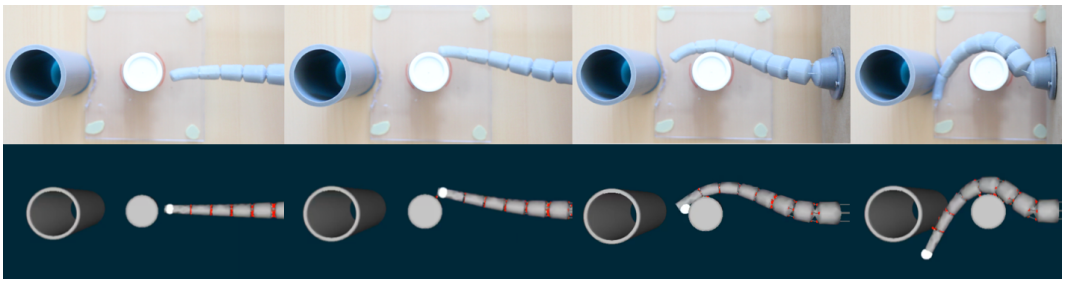

Figure 4.16: *Real cable-driven soft robot and the corresponding motion computed by the simulated inverse model. The input of the inverse model is the motion of the robot's tip.*

Having a real-time control of the robot motion could be particularly beneficial when dealing with evolving/changing environment. It would only require to update the obstacles position and/or shape in the simulation. Tracking the environment is a complex task. It could either be done with camera mounted on the robot, or with the help sensors.

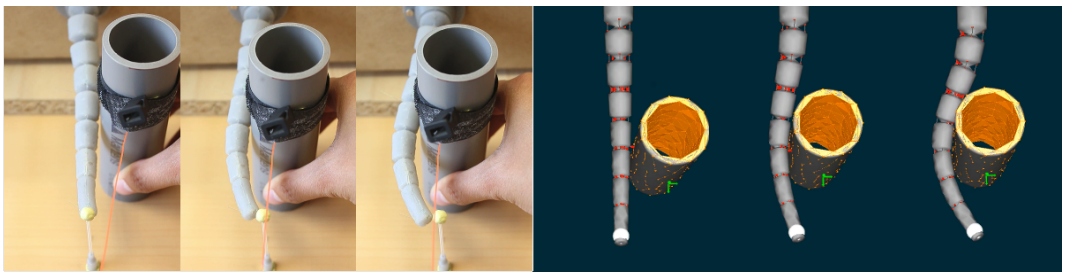

Figure 4.17: *Top: Real cable-driven soft robot, Bottom: Corresponding motion computed by the simulated inverse model. We interactively move an obstacle while the input of the inverse model is a fixed position of the robot's tip.*

In Figure 4.17, we show an example of the same robot but with a moving cylinder. We used a Gametrak device to interactively update the position of the moving obstacle in the simulation. In this scenario the target of the end-effector is fixed. Using our algorithm we are able to find the new configurations of the robot actuation so that the end-effector stays put, while "pushing" the trunk with the cylinder.

In Figure 4.18, we show some measurements made on the trunk end-effector position, on a 2D trajectory, when the course of the robot is disrupted by the obstacle. We obtained an average error of 3.6$mm$.

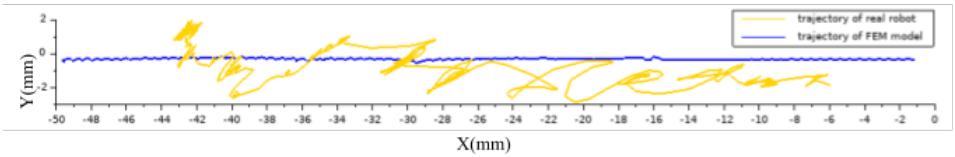


Figure 4.18: *2D trajectory of the real cable-driven soft robot end-effector, with corresponding trajectory of the FEM model. 3D average error of 3.6*mm.

### Performance

The two main computation steps of the simulation are the computation of the matrices $W_{ij}$ $(i, j = e, a, c)$ and the resolution of the sequence of QP problems.

In Table. 4.4, we show the average computation time[3] for these two main steps. We can see that the decomposition method we use to solve the contacts allows us to maintain interactive rates (between 30Hz and 100Hz). We recall that for more complex geometries, a large number of nodes may be required, and the size of the FEM matrix will consequently be large, making the computation of $W$ time consuming. If needed, we can use the work of (Courtecuisse et al. 2010) on asynchronous preconditioners, which gives real-time performance with around 6000 DoFs. For meshes with more than 6000 DoFs, we use the model order reduction method of Goury & Duriez (2018).

| Example | #DoFs | #Cont. | W | QPs | Sim. |
|---|---|---|---|---|---|
| Tower | 1680 | 21 | 10.53 $ms$ | 0.44 $ms$ | 36.40 $ms$ |
| Finger | 474 | 5 | 0.64 $ms$ | 0.08 $ms$ | 4.79 $ms$ |
| Beam | 480 | 19 | 3.38 $ms$ | 0.63 $ms$ | 13.17 $ms$ |
| Trunk | 1665 | 76 | 10.09 $ms$ | 4.91 $ms$ | 34.52 $ms$ |

Table 4.4: *Number of DoFs and the average number of contacts, computation time in ms of the matrices $W_{ij}$ construction, sequence of QPs resolution, and one time step of entire simulation.*

## 4.6 Applications

In this section we discuss two possible direct applications to our method. First, for the control of steerable catheter used for interventional radiology. Second, for an other medical application that does not involve a soft robotic system, but could still benefit from our method. That is for the registration of not directly attainable organs.

### 4.6.1 Catheter guidance

The optimization algorithm we propose can also be used for issues in interventional radiology, a minimally invasive way to operate vascular pathologies. The method consists in inserting therapeutic tools within the arteries, through a catheter which is a thin, flexible tube. This intervention is complex and requires good planning. The radiologist often uses fluoroscopic X-ray guidance to drive the instrument through the blood vessels. If the procedure takes time, the radiation load of both the operator and patient can increase. Numerous research projects aim at improving the manoeuvrability and controllability of catheters (Dupont et al. 2010), (Back et al. 2016), and advanced steering

[3]On a desktop computer with an i7 Intel processor 3.60GHz

concepts have been proposed by industry (Gould & Riggs 1986), (Martinelli 1997).

Using our method and a 3D model of the vessel network from images of the patient, we can imagine to provide a simulation of the intervention in the operating room. Our controller can help the physician to guide the tip of the catheter through the vessel network. A first simulation result is shown in Figure 4.19, where we control a soft catheter insertion into a forked tube.

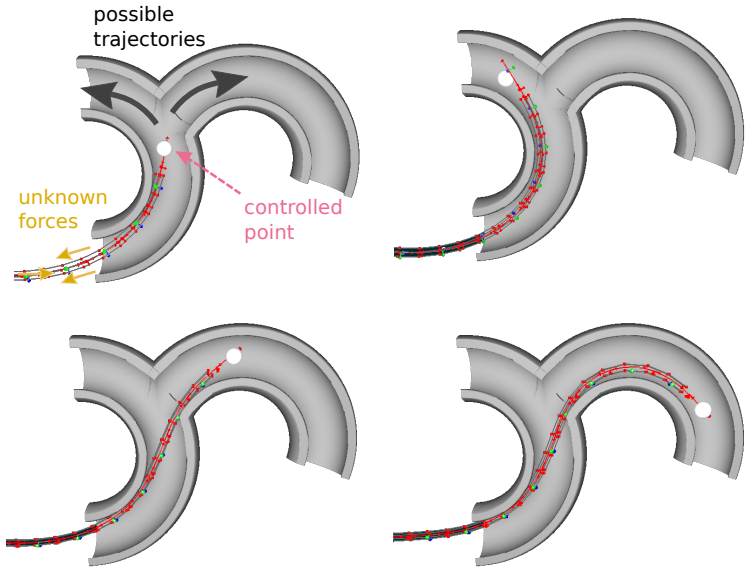


Figure 4.19: *Soft beam actuated with four cables. In this simulation we control the end-effector of a soft beam inserted in a pipe. The white sphere is the desired position.*

The rod has five actuators; four cables allow for orientation of the rod end-effector (up / down / left / right), and the fifth one moves the rod extremity along its longitudinal direction. Its design is a simplified version of the catheter described in (Ataollahi et al. 2016) (see Figure 4.20).

Using our controller, we are able to optimize the five actuators so that the rod tip can be inserted in both branches. Note that, for this application the inclusion of contacts in the inverse problem optimization is mandatory. We have started a collaboration with Back et al. to use our controller on their catheter and demonstrate the feasibility of the method. The catheter is modeled using co-rotational beam model and has 66 DoFs. The matrix $W$ was computed in an average of $0.75ms$, and the QPs in $3.13ms$ for an average number of contacts equal to 47.

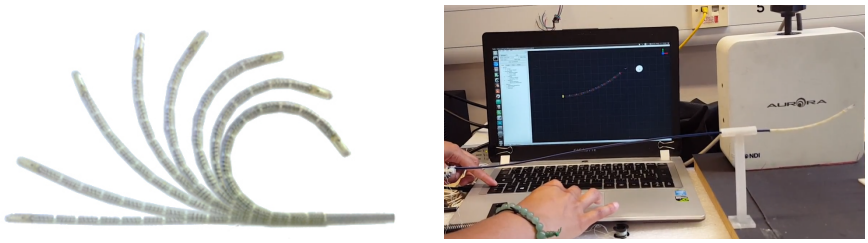

Figure 4.20: *(left) Catheter proposed in (Ataollahi et al. 2016). (right) Simulated version with control of the real device.*

### 4.6.2 Underlying organs position control

The method can also be investigate for the registration of unattainable organs. In Figure 4.21 we show a simulation of two colliding soft bodies (that can be identified as the skin and an organ for example), each body having some fixed part. The problem we want to solve in this simulation is how to push the wall of a first deformable body, using an instrument, so that we can control a point of a second deformable body, thanks to transmission of forces allowed by contacts. In this example, we show that our method can have possible extension in medical applications (like medical robotics): Indeed, for some medical applications, it could be useful to know how much force/displacement to apply on a deformable wall (for instance with ultrasound probe when doing robotic assisted echography) in order to obtain a desired motion on an underlying deformable organ. Of course, dedicated study would be necessary to validate the use of the method in such an application. Here it just demonstrates the genericity of our algorithm and a potential larger use than soft robotics. The number of DoFs of this simulation (combining both structures) is equal to 2313. The matrix $W$ was computed in an average of $10.45ms$, and the QPs in $0.12ms$ for an average number of 16 contacts.

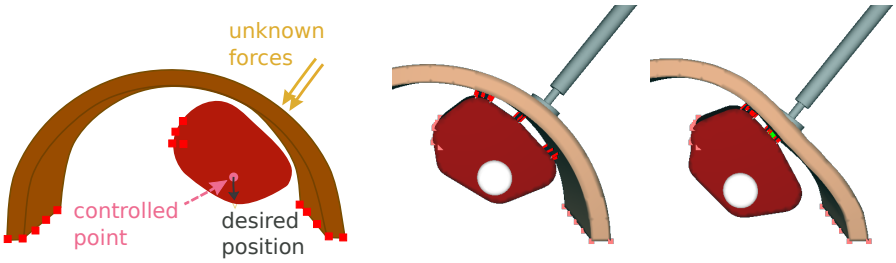


Figure 4.21: *Possible extension of the method: registration of colliding deformable bodies. The red squares show the fixed points. We control the position (white sphere) of a point of the red body by applying a force on the brown one.*

## 4.7 Conclusion

In this chapter we proposed to extend the optimization problem presented in chapter 3 to handle collision between the robot and its environment. As a first step we focused on non frictional contact, which can be sufficient for application when friction can be neglected. The problem is formulated as a QPCC due to the use of Signorini's complementarity condition for contacts. We proposed to use a specific solver based on decomposition method, to maintain the high computational rates of the inverse resolution. The method is compatible with dynamic environment, as long as the obstacles are modeled in the simulation and their position are updated. We discussed available tools for fast collision detection and presented numerous experiments showing the efficiency of the method, in terms of accuracy and computational time.

In the next chapter we present strategies employed to solve friction (with stick contact only) and open the control of some soft robot locomotion and grasping.

Chapter 5

# Inverse Model with Stick Contact Handling

## Contents

## 5.1 Introduction

Soft robots have been intensely investigated and developed for grasping applications. With traditional hard robot these tasks have always been complex to perform. It requires for example to have force sensor mounted on the robot to avoid damages, and the robot is often calibrated to grab specific object (size and shape). With soft robots these are no longer an issue. Companies like *Soft Robotics Inc*, propose soft robotic gripping systems for food and advanced manufacturing, as well as e-commerce. These kind of tasks are simple on and off system that do not suffer from precise control of the actuation. Here we are interested in the problematic of controlling the deformation or displacement/orientation of the grasped object.

The other task we are interested to control is locomotion, defined as the translation of the soft robot body from one location to another. In particular, we are looking at the control of locomotion based on rolling. We again want to provide real-time solution to able a control in moving environment.

In the following we describe how we extend the method proposed in the latter chapter to handle static friction (i.e. stick contact, without sliding effect). The extension is simple yet opens the control of much more complex tasks. In the following sections 5.2 and 5.3, we detail the formulation of the IP with sticking contact, and the solver we propose to solve the problem in real-time. In sections 5.5 and 5.6, we respectively present results of locomotion control (rolling), and grasping control, with several numerical examples and one real experiment.

## 5.2 Formulation of the IP with sticking contact

Coulomb's law is usually used to model the effects of sticking and sliding contacts. The contact force is thus given by the composition of a normal force $H_n^T\lambda_n$ and a tangential (friction) force $H_t^T\lambda_t$, given $H_c^T\lambda_c = H_n^T\lambda_n + H_t^T\lambda_t$. The Coulomb's model imposes the contact force to lie in a circular cone, called friction cone. If the force is (strictly) inside the cone, the contact is sticking. If the force is on the cone boundary, the contact is sliding.

Let $\mu$ be the friction coefficient that gives the degree of adhesion of the contact. Coulomb's law imposes that one of the three following cases must occur:

$$\begin{cases} \text{either:} & \lambda_n = 0 \text{ and } \delta_n \geq 0 & \text{(inactive)} \\ \text{or:} & ||\lambda_t|| < \mu\lambda_n \text{ and } \delta_c = 0 & \text{(sticking)} \\ \text{or:} & ||\lambda_t|| = \mu\lambda_n \text{ and } \delta_n = 0 & \text{(sliding)} \end{cases} \tag{5.1}$$

The circular cone leads to a Non-Linear Complementarity Problem (NLCP) which is difficult to solve numerically. Therefore the usual approach in treating this problem has been to formulate a LCP with a polygonal approximation of the friction cone (Anitescu & Potra 1997). This problem can then be solved (for the forward case) using for example Lemke's algorithm.

However, the global problem of solving both the actuation and friction contact (in the inverse manner) is more complex, and solving this problem in real time is very challenging. To simplify the resolution, we make the assumption of sticking contact only, which can be sufficient for the control of some soft robots locomotion or for manipulation tasks.

Let us note $n_a$ and $n_c$ the number of actuator and contact forces. We formulate the constraints on contact as follow. Given a contact $c$:

$$\begin{cases} \text{either:} & \lambda_n = \lambda_t = 0 \text{ and } \delta_n \geq 0 \quad \text{(inactive)} \\ \text{or:} & \lambda_n \geq 0 \text{ and } \delta_n = \delta_t = 0 \quad \text{(sticking)} \end{cases} \tag{5.2}$$

Let us recall that, to control the robot motion, we want to find the actuation $\lambda_a$ so that effectors reach their desired positions. This corresponds to minimizing the norm $\|\delta_e\|$, while respecting the constraints (5.2), yielding to the following QPCC:

$$\begin{aligned} \min_{\lambda_c, \lambda_a} \quad & \left\| W_{ec}\lambda_c + W_{ea}\lambda_a + \delta_e^{\text{free}} \right\|^2 \\ \text{s.t.} \quad & (5.2) \end{aligned} \tag{5.3}$$

## 5.3 Specific solver

For frictionless contact, we proposed in chapter 4, a specific solver to handle the complementarity constraints introduced by Signorini's law. The algorithm was based on decomposition method. The same approach can be used here to solve the new QPCC (5.3). We remind that the main idea of the method is to use the disjunctive structure of the complementarity constraints to formulate series of QP. At each iteration of the QPCC solver, we specified two subsets $\mathrm{I}_1$ and $\mathrm{I}_2$ with $\mathrm{I}_1 \cup \mathrm{I}_2 = \{1, \ldots, n_c\}$, respectively distinguishing the inactive and active (sticking) contacts, yielding:

$$\begin{aligned} \min_{\lambda_a, \lambda_c} \quad & \left\| W_{ea}\lambda_a + W_{ec}\lambda_c + \delta_e^{free} \right\|^2 \\ \text{s.t.} \quad & A\lambda_a \geq b \\ & (\lambda_n)_i = (\lambda_t)_i = 0 \text{ and } (\delta_n)_i \geq 0, && \text{for } i \in \mathrm{I}_1 \\ & (\lambda_n)_i \geq 0 \text{ and } (\delta_n)_i = (\delta_t)_i = 0, && \text{for } i \in \mathrm{I}_2 \end{aligned}$$

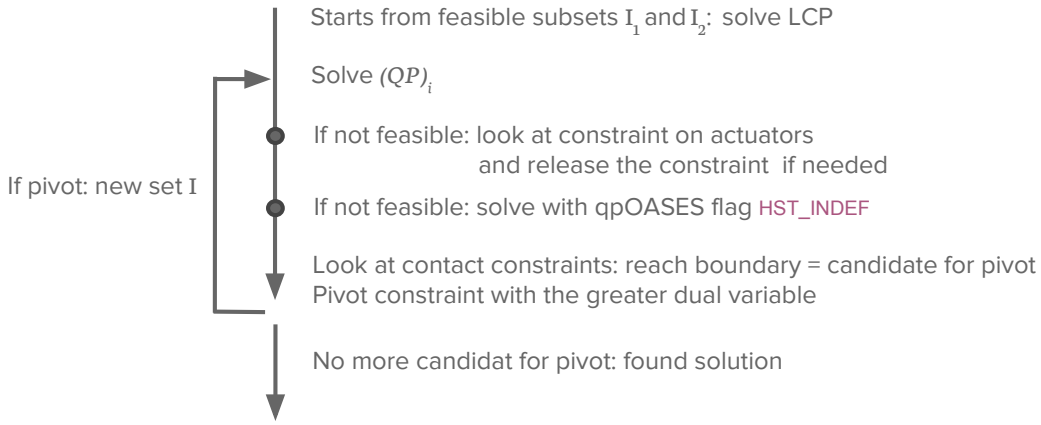


Figure 5.1: *Strategy scheme to solve the inverse problem with friction contact handling.*

For the decomposition method to converge, we said that the iteration has to start from initial feasible subsets $I_1$ and $I_2$. To find these feasible subsets in case of friction, we again consider the actuation fixed. For this stage, either the actuation computed during the previous time step of the simulation is available (warm start) or we set the actuation to be equal to zero. With the actuation fixed, the contact problem is expressed as a NLCP. We use a Gauss Seidel algorithm to solve it. In this resolution, we allow contacts to lie outside the friction cone to obtain strictly sticking contacts.

When there is no more candidate for pivot the iterations stop, and the solution, which can be a local minimum, is the one given by the last QP resolution. A scheme of the solver is given in Figure 5.1.

## 5.4 Well-posed problem

The addition of friction contact into the optimization problem raised new issues, that we noticed from experiments. In addition, the assumption of strictly sticking contact in the IP may lead to ill-posed problems. In this section, we discuss some problems that have been identified.

### 5.4.1 Infeasible constraints

We noticed, from experiments with locomotion, that some constraints on actuators may become infeasible, due to a motion induced by contacts and the dynamic of the robot. For example, it is assumed, in the simulation, that cables are inextensible and straight. Yet, if the robot enters a contact with certain velocity (see Figure 5.2), it may happen that the motor mounted on

the real robot does not pull the cable fast enough (to ensure that the cable remains straight), or that the course of the cable is already at its maximum. The corresponding constraints in the optimization problem (i.e. limits on cable maximum/minimum displacements and/or cable displacements variation) may no longer be satisfied.

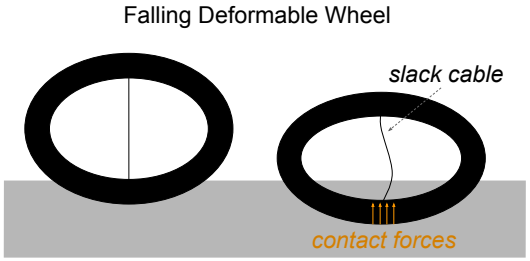


Figure 5.2: *Example where a cable can not remain straight. In such cases, the cable constraint (i.e. limit on displacement) is removed from the optimization problem, for a certain amount of time that corresponds to the time it takes to the motors to pull on the cable.*

In the implementation, when such cases are detected, we notify the user and we temporarily remove the corresponding constraints from the optimization problem.

### 5.4.2 Grasping constraint

For manipulation tasks, we want to control the motion of the grasped object, instead of the gripper itself (see section 5.6 for examples). Yet, if the squeezing force (the real gripper applies on the object) is not sufficient, the assumption of sticking contacts we make in the simulation, may not be satisfied in the real world; sliding contacts may appear between the real gripper and the object. Also, in some configurations, dropping the object may lead to a better solution at a given time step; usually when the target is far from the grasped object we want to control.

Thus, to prevent sliding contacts on the real robot, and also to prevent the gripper from releasing the object when possible, we need to constrain the gripper tightening. To this end, we insert into the optimization problem an additional constraint on the contacts force.

The simplest formulation is to constrain the sum of the contacts force (along the normal direction) to be greater than a given value, yielding:

$$\mathbb{I}^T \lambda_c \geq \lambda_{c,min}$$

with $\mathbb{I}$ a column vector of zeros and ones, with $\mathbb{I}_i = 1$ if $i$ in the set of the contacts normal indices. To estimate a good value for $\lambda_{c,min}$, we simply run a simulation where the object is being hold against the gravity and store a corresponding range for the product $\mathbb{I}^T\lambda_c$. The configuration is easily obtained by having the effectors target located in the robot working space.

### 5.4.3 Indefinite Hessian

It appears sometimes that the matrix of the QPCC is no longer positive definite, nor positive semi-definite (e.g. when the size of the effectors space is too low), but indefinite. The cause of this problem has not been identified yet, while the problem appears only in some experiments with friction. However, when it is detected, we use the implementation provided by qpOASES for indefinite Hessian, when solving a $\text{QP}_\text{I}$, piece of the global QPCC problem. Unfortunately in that case, the convergence of the algorithm is no longer guaranteed. In practice, the solver finds a solution that is not far from the previous one (i.e. the solution computed at the previous time step), but can either way produces small oscillations.

## 5.5 Locomotion

Numerous research projects aim at designing soft robot able to walk (Tolley et al. 2014), crawl (Sugiyama & Hirai 2006), or roll (Steltz et al. 2009).

The generation of walking pattern for legged robots has been a focus of many studies from the traditional robotic community (Huang et al. 2001) (Wieber et al. 2016). It involves to take stability into account, to be able to walk in different environments, such as on uneven terrain, up and down slopes, or in regions with obstacles. Even though work on legged robots have lead to incredible results on making robots able to walk and run dynamically through rough terrain (Raibert et al. 2008) (Hyun et al. 2014), soft robotics is an opportunity to rethink locomotion. It enables simple design and control to generate mobility.

Like in (Lee et al. 2013), where Lee et al. propose a deformable wheel robot using origami structure. The wheel can change its shape from a long cylindrical tube to a flat circular tube using only few actuators. Using soft materials has several interest such as adapting the robots size and squeezing to small spaces, adapting their shape to obstacles, or navigating through sensitive environment. The work proposed by Hawkes et al. (2017), on soft robot navigation through growth (inspired by plants), is also interesting as the robot evolves without using friction to power the motion, nor sliding though its environment. This

kind of navigation has numerous advantages, it is for example better suited for sticky surfaces and going through air to pass a ditch, as demonstrated in their paper. However, usual locomotion is maybe more adapted to dynamic environment.

We conducted numerical experiments on two designs of soft robots able to roll, that we present in this section. Both designs were inspired by soft robots proposed by other labs. While the actuations we modeled is not exactly the same as their (e.g. cable instead of SMA, and Young's modulus optimization instead of jamming), the presented numerical experiments are showing really good results.

### 5.5.1 Circular soft robot

In (Sugiyama & Hirai 2006), Sugiyama & Hirai propose a soft circular robot actuated with eight shape-memory alloy coils capable of crawling and jumping (see Figure 5.3). To control the robot, they define by hand a periodic voltage pattern that produces the motions. Based on their design, we model a circular soft robot actuated with four cables (see Figure 5.4).

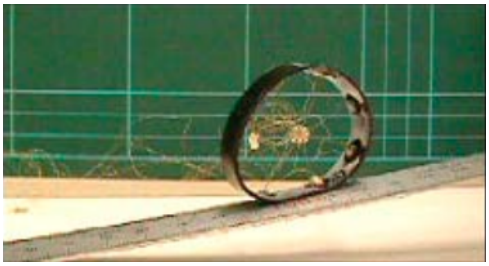

Figure 5.3: *Soft robot proposed by Sugiyama & Hirai (2006) (image from their paper). The robot is actuated with eight SMA coils.*

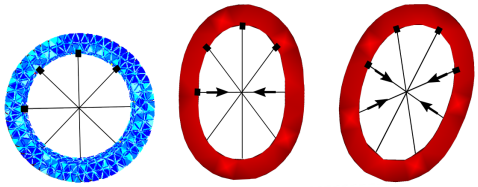

Figure 5.4: *Circular cable-driven soft robot. (left) Rest shape and FEM mesh. (middle, right) Deformed shape with corresponding actuation.*

Using our framework, we built a simulation in which we are able to drive the circular robot on a slope (see Figure 5.5), by driving the position of its barycenter. Simply moving the barycenter target forward (or backward) makes the deformable structure start to roll and reach the desired position. As the method has real-time performance, we could track the real terrain steepness and update the simulation consequently.

We can also make the robot move forward while controlling/adjusting its height, enabling it to pass under obstacle (see Figure 5.6). Note that, for this example,

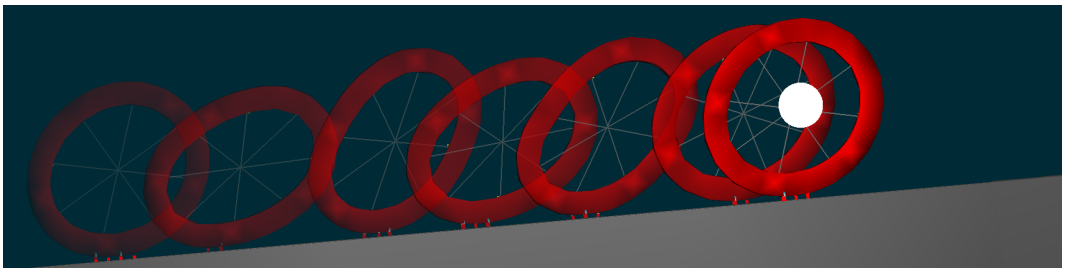

Figure 5.5: *Simulation of the circular robot locomotion. The algorithm finds how to actuate the four cables to control the position of the deformable structure barycenter. Moving the barycenter target (white sphere) forward or backward makes the structure roll.*

at each time step of the simulation, the QPCC solver finds a solution in one or two iterations.

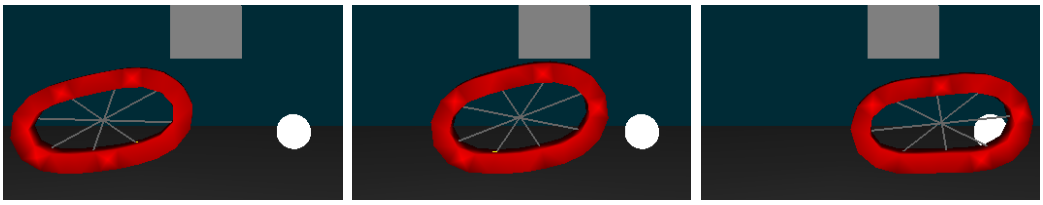

Figure 5.6: *Simulation of the circular robot locomotion. The algorithm finds how to actuate the four cables to control the position of the deformable structure barycenter, and make it pass under an obstacle.*

### 5.5.2 Spherical soft robot

In (Steltz et al. 2009), Steltz *et al.* propose a spherical robot composed of cells around its outer perimeter (each cell being filled with jamming material), and a centeral actuated cavity (see Figure 5.7). By unjamming a subset of outer cells, and inflating the central actuator, the robot is capable of rolling. We modeled a similar spherical robot, an icosahedron with a central cavity (see Figure 5.8).

We were able to optimize both the stiffness (Young modulus) of each cell and the pressure to apply in the central cavity to drive the soft structure (see Figures 5.9). To control the robot displacement, we optimize the position of its barycenter, and two additional points located on its surface. To make the robot roll, we move the desired position of the barycenter and compute the kinematic of the corresponding rigid sphere to obtain the target of the two other controlled points. In comparison with the circular robot example, we

Figure 5.7: *Spherical soft robot proposed by Steltz et al. (2009, 2010) (image from their paper). The robot is composed of 20 cells around its outer perimeter (each cell being filled with jamming material), and a centeral actuated cavity.*

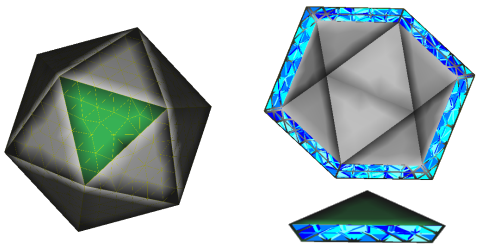

Figure 5.8: *Spherical soft robot. (left) Global shape with one of the 20 cells shown in green. (top right) Cross-section of the robot showing the central cavity and the FEM mesh. (bottom right) One isolated cell. Note that the FEM mesh follows the cells boundary.*

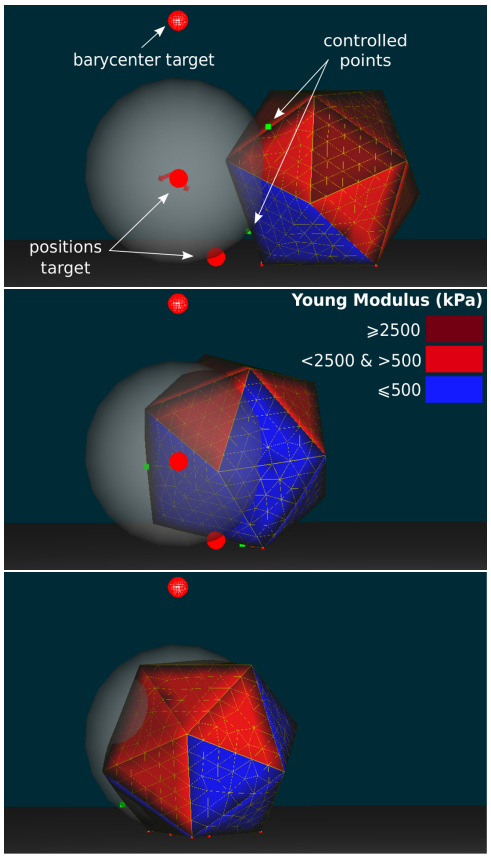


Figure 5.9: *Simulation of the spherical robot. The inverse problem outputs a different Young modulus for each cell, and a pressure for the inner cavity to make it roll to a target position.*

have to control more points than just the barycenter; the two additional points allow us to describe a global movement.

## 5.6 Grasping and manipulation

For grasping tasks, we propose to control the position/orientation of the grasped object instead of directly controlling the position of the soft robot.

Grasping tasks involve three steps (see Figure 5.10): first to reach the object, second to grasp the object, and third to control the grasped object. We already demonstrated that reaching the object can be done efficiently using our controller, even in presence of obstacles. The second step is more difficult to

handle: we have to find how to grasp the object, that is choose the right contact points, the right configuration of the gripper to take the object. Depending on the geometry and motion of both robot and object, this task can be complex. In this section we are interested in the third step, that is controlling the object when it is hold. In this configuration, the assumption of sticking contact only can be satisfied.

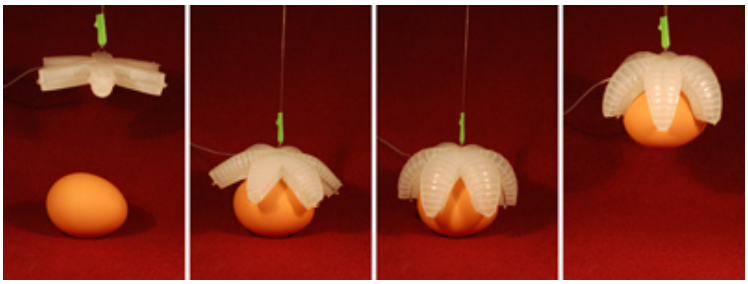

Figure 5.10: *Starfish gripper from (Ilievski et al. 2011), illustrating the three steps of a grasping task, and power grasp.*

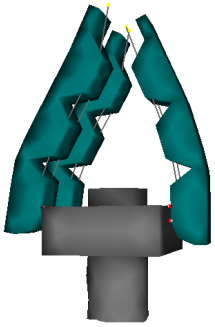

Figure 5.11: *Simulation of finger grasping.*

When the robot is holding the object, we can distinguish two types of grasp. First, power grasp, like in the starfish example: the motion of the object is directly linked to the motion of the robot. This is more simple as we do not have to consider the model of the object. The second type of grasp, called precision grasp, is harder to control: the motion of the object is decoupled from the motion of the robot, and it requires to have a model of the object (see Figure 5.11).

Here we are interested in the latter type of grasping; either to control the grasped object position and orientation, or to control the object deformation. This kind of control requires the modeling of both gripper and object. In the following we show results of the two scenarios.

### 5.6.1 Soft trunk gripper

For this first experiment, we modified a bit the shape of the soft trunk proposed in (Coevoet, Escande & Duriez 2017) to make it able to grab objects. The new shape is a bit longer and have higher low-cut joints. In this scenario, the trunk is holding a cup, and is making it move in position and orientation. In the simulation both the cup and the trunk are soft. In Figure 5.12 we show results of the simulation with different global position and orientation target for the

cup. In Figure 5.13 we control the orientation of a real plastic cup online using our simulation framework.

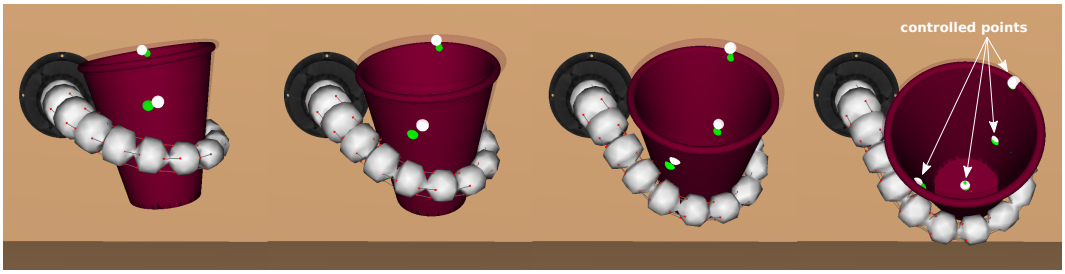


Figure 5.12: *Simulation of a soft gripper holding a deformable cup subject to gravity. Here we optimize the cables displacements to control the position/orientation of the cup. A phantom of the cup target is shown in transparency. The cup have four controlled points represented by the green spheres. The corresponding targets are represented by the white spheres.*


Figure 5.13: *Real soft robot actuated online using the output of the simulation. In this scenario, using our control framework, we are able to control the orientation of the real plastic cup (see the attached video).*

In Figure 5.14, we give the trajectories of the plastic cup center of mass. A magnetic sensor was placed at the bottom of the cup. The sensor gives both the position and the orientation of the real cup. Measured error for the entire animation are given in Table 5.15 (please see the video for better understanding). We see that with this open-loop system we can obtain good match between the simulation and the robot.

Yet, the robot could be equipped with sensors to correct the model on the tightening force. Indeed, if this force is not sufficient, the real cup may slide. Note that if the target is out of the robot working space, the algorithm will try to optimize the displacement to fit at best the target, but within the space of possible configurations. For this example, at each time step of the simulation, the QPCC solver finds a solution in an average of seven iterations.

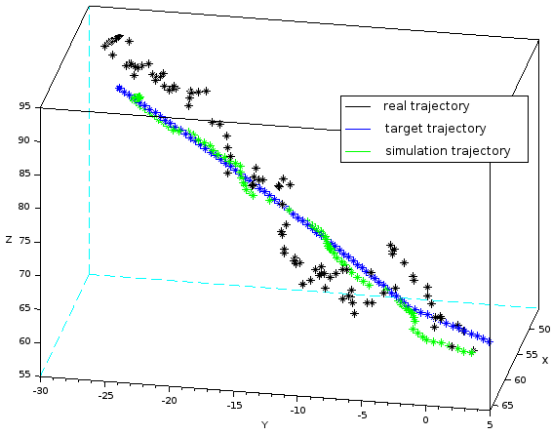


Figure 5.14: *Trajectories of the plastic cup center of mass: real trajectory from the magnetic sensor, target trajectory and simulation trajectory (i.e results from the trunk actuation optimization).*

| | Target vs. Sim. | |
|---|---|---|
| | mean | stdev |
| c.o.m | 1.34 *mm* | 0.37 *mm* |
| orientation | 2.21 *deg* | 1.60 *deg* |

| | Sim. vs. Reality | |
|---|---|---|
| | mean | stdev |
| c.o.m | 8.69 *mm* | 3.79 *mm* |
| orientation | 5.79 *deg* | 3.58 *deg* |

Figure 5.15: *Mean error and standard deviation of the plastic cup center of mass and orientation (for the trajectories given in Figure 5.14). The error of the orientation is calculated on the plan of the greater displacement (see Figure 5.13).*

### 5.6.2 Grasping and manipulation of deformable objects

We propose again to control the grasped object instead of directly controlling the position of the robot. In this example we use our algorithm to control not only the position, but the shape of a structure being deformed by a rigid hand (the anthropomorphic Schunk 5-finger hand grips [1], see Figure 5.16).

For each of the three fingers we optimize, we consider only the tip. Each tip has three DoFs (three translations). Thanks to the transmission of forces allowed by contacts, we are able to control the fingers tip position so that the deformable object reaches a desired shape. The kinematics of the hand articulations are derived by the displacement of the tips. The deformable structure has no fixed part and is, like in the other simulations, subject to gravity. We control four positions on the deformable object surface (see Figure 5.16). For this example, at each time step of the simulation, the QPCC solver finds a solution in one or two iterations. A first improvement would be to optimize directly each articulation of the fingers, instead of the tip position. This way we would constrain the articulated chain, to allow the generation of plausible motions, and deal with singularities in the fingers. For instance, having the articulations aligned is often avoid in rigid robotics, because it induces multiple actuations

[1] The Schunk hand CAD model was provided by the manufacturer.

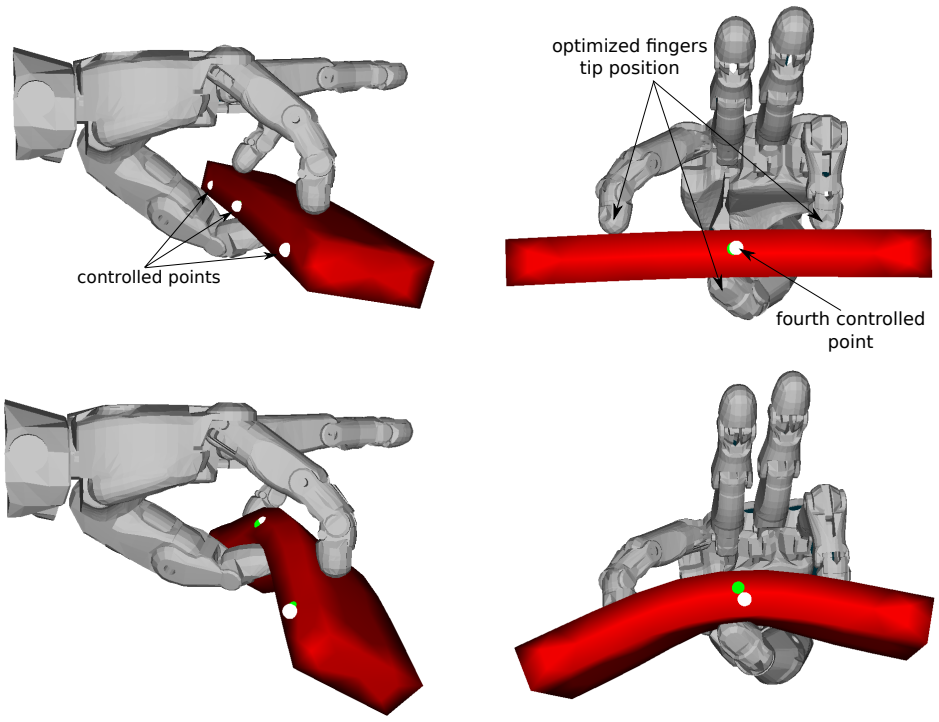


Figure 5.16: *Simulation of a soft beam subject to gravity, and held and deformed by three rigid fingers. The optimization finds the fingers tip position that leads to a desired deformation of the beam, and also prevents it from falling. The controlled points are the green spheres and their target are the white spheres.*

that lead to the same tip position.

This example raised from a collaboration with Ficuciello et al., where in (Ficuciello et al. 2018), we first propose to use our inverse algorithm, without considering contact, and in a decoupled fashion, for the same application. The resolution is decomposed as follow: a first simulation evaluates the force to apply on three fixed regions of the soft object, to deform it as desired. The regions are delimited by circles, the intensity of forces is distributed from extremity (low intensity) to center (high intensity) of circle, and applied in opposition to the surface normal direction. Note that this distribution of force intensity is known in the inverse resolution (i.e. stored in the matrix $H_a$ which contain the direction of the forces). The second simulation is running the kinematics of the whole hand, by applying the new position to the three fingers tip, computed in the first simulation. The third and last simulation is the resolution of the forward problem when applying the hand kinematics and considering friction contact with the grasped object (while considering sticking

contact only, see Figure 5.17). The three simulations are running at same time and are sending datas to each other in a request-reply pattern [2]. Thus the three simulations are running at the rate of the slowest one.

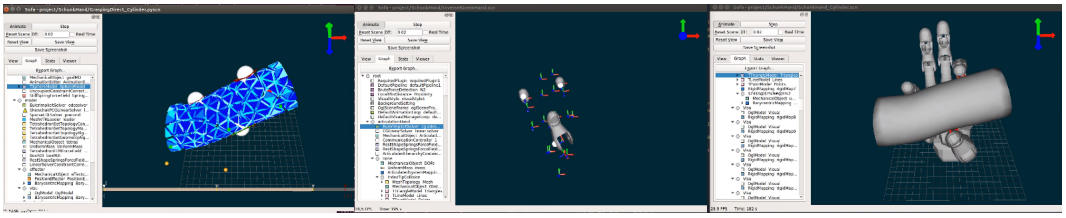

Figure 5.17: *Same scenario but in a decoupled fashion. The object has an extremity fixed.*

The best approach, in terms of correctness (simulate and control the reality) and stability (dealing with singularity problems), is to consider friction in the IK and optimize the articulated chain.

## 5.7 Performance

In this section, we give the computation time[3] of each simulation shown in this chapter. The two main computation steps of the simulation are the computation of the matrices $W_{ij}$ $(i, j = e, a, c)$ and the resolution of the sequence of QP problems. In Table. 5.1, we show the average computation time for these two main steps.

| Examples | #Cont. | #DoFs | W | QPs | Sim. |
|---|---|---|---|---|---|
| Circular | 9 | 2238 | 7.91 $ms$ | 0.21 $ms$ | 38.51 $ms$ |
| Spherical | 39 | 3003 | 79.56 $ms$ | 10.67 $ms$ | 151.29 $ms$ |
| Fingers | 12 | 1335 (beam), 9 (fingers) | 2.87 $ms$ | 0.98 $ms$ | 15.80 $ms$ |
| Trunk (GPU) | 81 | 2127 (trunk), 3765 (cup) | 76.53 $ms$ | 35.47 $ms$ | 185.18 $ms$ |

Table 5.1: *Number of DoFs and the average number of contacts, computation time in ms of the matrices $W_{ij}$ construction, sequence of QPs resolution, and one time step of entire simulation.*

For the trunk and the cup simulation, there is a high number DoFs. In such case we can use a GPU based algorithm (Courtecuisse et al. 2010) to compute

[2] Through a simple local communication protocol allowed by the ZeroMQ library. The implementation is available in the SoftRobots plugin.

[3] On a desktop computer with an i7 Intel processor 3.60GHz and a NVIDIA NVS 510

the matrix $W$. Using this approach we were able to run the simulation at near real-time.

## 5.8 Conclusion

In this chapter, we propose a generic algorithm to control the motion of soft robots able to grasp objects or roll from one place to another, with the assumption that each contact is either sticking or inactive. The method has real-time performance, which allow us to interactively control our robots and deal with evolving environments, as long as they are known.

The method is mainly limited by the fact it does not handle dynamic friction (sliding contacts). Thus, we should only use this method on scenarios where sliding effects are negligible. As mentioned in the section 5.6, in the soft trunk example, the robot does not grab the cup well enough to always prevent it to slide. We think that this problem should vanish by either improving the robot design or using a closed-loop strategy with sensor mounted on the robot to correct the model on the tightening force. As a future work, we would also like to test our method on a real gripper, and work on a pipeline to be able to switch from a control of the end-effector to a control of the grasped object.

CHAPTER 6

# Conclusion

## Contents

## 6.1 Summary and assessment

With this thesis we proposed, for the first time, generic methods (i.e. without any particular assumptions on the geometry) to inverse the model of soft robots. These methods are closed to what is used in traditional robotics to inverse rigid robots model, allowing to make a bridge - until now nonexistent - between the tools used in both communities.

The differences being that, our methods are based on the FEM to capture the deformations of the robot's structure and of its environment when deformable. Yet, as largely employed with their rigid counterpart, we formulate the problem of their inverse kinematics and dynamics as optimization programs. Allowing easy handling of constraints on actuation and singularity problems.

We are able to control several types of actuation, such as cable, pneumatic and hydraulic actuations, but also Young's modulus variation and motors directly attached to the deformable structure. Moreover, most of the applications involve interaction of the robot with obstacles. In that matter, we proposed new methods that include contacts into the optimization process. This method makes an important step for soft robotics, as we think that the knowledge of contacts in the modeling is all the more important. Indeed, soft robots kinematics is highly dependent on environmental factors. Finally, we proposed to control some soft robots during locomotion and manipulation tasks, which require the use of contact with static friction (sticking contact).

As usually the case in traditional robotics, we gave a particular attention to provide solutions with real-time performance. Allowing online control of soft robots in evolving environments.

### 6.1.1 Validation

As every simulation based approach, careful validation steps have to be taken for each robot modeled and simulated. The behavior of any simulated robot is sensitive to many properties such as the mechanical parameters, the mesh discretization, and boundary conditions etc. Since our actual robots are proofs of concepts, they do not meet industry criteria of quality control. In order to estimate the modeling and numerical errors that are aggregated in the simulated model, our approach consists in using test-benches composed of tracking devices. These devices (usually IR camera and markers) track the behavior of the real robots moving in all their working space and are compared with their digital counterparts. For instance, the soft robots (called "Diamond robot" in (Coevoet, Morales-Bieze & et al. 2017) and "Trunk" presented in section 3.4.3) have been validated using this methodology. However, this work

has not yet been conducted systematically.

### 6.1.2 Performances and accuracy aspects

Several aspects may have an impact on the performance of the simulation. Most significant being the spatial discretization (number of elements) as well as the materials constitutive equations. As our framework is usable either for interactive simulation or to control real soft robots, special care has to be taken depending on the final use of the simulation. Depending on the application, the level of accuracy that is needed, varies.

For instance, when the user is in the control loop, and the inverse simulation is there to help; as long as the general effect of the actuators on the robot's motion is predicted correctly, small errors in displacement can be tolerate. Indeed, the user will adapt his movement to succeed in reaching his goal.

In other cases, when the robot adapts its position according to the contacts it undergoes with the environment, the accuracy level needed in displacement is much higher. Indeed, in this configuration, large errors in displacement can lead to poorly detect contacts and poorly predict the correct motion of the robot.

## 6.2 Future work and perspectives

In this section we discuss the future work and perspectives of the thesis.

### 6.2.1 Locomotion and grasping

As for now, we are able to control two of the three steps of grasping. That is, reaching the object, and controlling it when held. Yet, we miss a control strategy for grasping the object, as it often induces sliding contacts with friction effects. Two possible directions can be taken, either we succeed in extending the algorithm to dynamic friction (sliding), or we could investigate on/off control strategies for grasping (which can be possible depending on the device). When the object is held, we could now make the simulation switch between the control of the gripper and the control of the object.

Finally, no experiment has yet been conducted on locomotion with a real robot. As a future work, we would like to test our algorithm on the control of a real soft robot able to roll, for instance.

### 6.2.2 QPCC solver performance

The performance of the QPCC solver, in terms of computation time, highly dependents on the size of the problem. This size is given by the number of actuators and contacts. The number of actuators is usually low (less than 20), while the number of contacts can be large and degrade the performance of the solver.

Solving a large QP system can be expensive, thereby, the decomposition method used to solve the QPCC, which can generate sequence of QPs, can be even more expensive. Two approaches were suggested in the thesis to deal with this issue:

- First, a reduced formulation, that fully expresses the contact force $\lambda_c$ with respect to the actuation force $\lambda_a$, and thus produces a sequence of QPs with the size of the actuation variables. For the moment, the reduced formulation has only been implemented and tested on Matlab. Note that extend the approach to friction contacts is straight forward. Indeed, in case of static friction (stick contact), equation (4.19) for an active contact becomes:

$$\begin{bmatrix}\lambda_n\\ \lambda_t\\ \lambda_s\end{bmatrix} = -\bar{S}\left(\bar{S}^T\begin{bmatrix}W_{nn}W_{nt}W_{ns}\\ W_{tn}W_{tt}W_{ts}\\ W_{sn}W_{st}W_{ss}\end{bmatrix}\bar{S}\right)^{-1}\bar{S}^T\left(\begin{bmatrix}W_{na}\\ W_{ta}\\ W_{sa}\end{bmatrix}\lambda_a + \begin{bmatrix}\delta_n^{free}\\ \delta_t^{free}\\ \delta_s^{free}\end{bmatrix}\right)$$

  The only difference here being the size of the matrix to inverse (three times larger than without friction). As a future work we would like to implement the algorithm in C++ and conduct tests for the friction case.

- The second approach, is to use a detection method that reduce the number of contact points. We already mentioned the contact detection method based on layer depth images, which can reduce both the computation time of the detection phase, and the number of contact points needed to solve the inter-penetrations. However, the experiments we conducted with the method proposed by Allard et al. (2010) often yielded to unstable simulation. That is, for an actuation that we fix, one can observe continuous changes on the detected contact points. The origin of this instability has not been identified yet.

### 6.2.3 Closed-loop and control laws

As any control method based on a model, the approaches presented in this thesis strongly rely on the quality of the underlying modeling.

The results shown in this thesis are open-loop experiments. There is no feedback signal from the real soft robots to correct and adapt the inverse resolution outputs. This makes the robots sensitive to untracked external disturbances, and is known to reduce the accuracy of the motion control. While we have shown results with low errors in general, these errors could be however reduced using a feedback controller.

A first attempt is to introduce a visual servoing control method (Zhang et al. 2016). In this approach, the robot is simulated in real-time and an observer make sure that the configurations of both the real robot and its simulation model stays very close. The method has not been tested yet on the algorithms proposed in this thesis. The difference being the use of optimization methods to inverse the model.

A limitation of this closed-loop control strategy is that, it is based on a quasi-static model of the robot, while neglecting the dynamic part of the model. Yet, the dynamics can have high effects depending on the robots. In that matter, one can consider the use of dynamic models in the control laws, as propose in (Thieffry et al. 2017). However the state of these works is very preliminary and not yet compatible with the inverse modeling strategies detailed in this thesis.

### 6.2.4 Other perspectives

In this thesis, we proposed algorithms to optimize soft robots actuation, over single time-step, and to solve predefined trajectories. Yet, for some applications, it would be beneficial to optimize the trajectory path, and/or optimize the actuation over multiple time-steps.

For example in (Bretl et al. 2005), Bretl et al. propose a method to find a path for a wall climbing robot, while satisfying certain constraints (e.g. equilibrium, collision, joint torque limits). For this kind of tasks, one must plan an entire sequence of steps, where each one might have future consequences. This process is called multi-step planning, and is used for general multi-limbed locomotion and manipulation planning.

It would be very interesting to invest such techniques and try to apply them to soft robotics.

# List of Publications

## Journal articles

[Coevoet, Morales-Bieze & et al. 2017] Coevoet E, and Morales-Bieze T, & et al. (2017), 'Software toolkit for modeling, simulation, and control of soft robots', *Advanced Robotics* 31(22) **(Best paper award)**.

[Coevoet, Escande & Duriez 2017] Coevoet E, Escande A, & Duriez C (2017), 'Optimization-Based Inverse Model of Soft Robots With Contact Handling', *IEEE Robotics and Automation Letters (Proc. ICRA)* 2(3).

[Coevoet et al. 2015] Coevoet E, Reynaert N, Lartigau E, Schiappacasse L, Dequidt J, & Duriez C (2015), 'Registration by interactive inverse simulation: application for adaptive radiotherapy', *International Journal of Computer Assisted Radiology and Surgery* 10(8).

## Conference papers

[Coevoet et al. 2018] Coevoet E, Escande A, & Duriez C (2018), 'Soft robots locomotion and manipulation control using FEM simulation and quadratic programming', *in* 'IEEE International Conference on Soft Robotics'.

[Ficuciello et al. 2018] Ficuciello F, Migliozzi A, Coevoet E, Petit A, & Duriez C (2018), 'FEM-based deformation control for dexterous manipulation of 3D soft objects', *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems'.

[Rodríguez et al. 2017] Rodríguez A, Coevoet E & Duriez C (2017), 'Real-time simulation of hydraulic components for interactive control of soft robots', *in* 'IEEE International Conference on Robotics and Automation'.

[Duriez et al. 2016] Duriez C, Coevoet E, & et al. (2016), 'Framework for online simulation of soft robots with optimization-based inverse model', *in* 'IEEE

International Conference on Simulation, Modeling, and Programming for Autonomous Robots'.

[Largilliere et al. 2015] Largilliere F, Verona V, Coevoet E, Sanz-Lopez M, Dequidt J & Duriez C (2015), 'Real-time Control of Soft-Robots using Asynchronous Finite Element Modeling', *in* 'IEEE International Conference on Robotics and Automation'.

[Coevoet et al. 2014] Coevoet E, Reynaert N, Lartigau E, Schiappacasse L, Dequidt J, & Duriez C (2014), 'Introducing Interactive Inverse FEM Simulation and Its Application for Adaptive Radiotherapy', *in* 'Medical Image Computing and Computer-Assisted Intervention'.

# List of Figures

# List of Tables

# Bibliography

[Allard et al. 2010] Allard J, Faure F, Courtecuisse H, Falipou F, Duriez C & Kry P. G (2010), 'Volume contact constraints at arbitrary resolution', *ACM Transactions on Graphics* 29(4). 92, 94, 104, 135, 143, 149

[Alliez et al. 2005] Alliez P, Cohen-Steiner D, Yvinec M & Desbrun M (2005), 'Variational Tetrahedral Meshing', *ACM Transactions on Graphics* . 71, 149

[Anitescu & Potra 1997] Anitescu M & Potra F. A (1997), 'Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems', *Nonlinear Dynamics* 14(3). 118, 149

[Aristidou & Lasenby 2011] Aristidou A & Lasenby J (2011), 'Fabrik: A fast, iterative solver for the inverse kinematics problem', *Graphical Models* 73(5). 41, 42, 149

[Aristidou et al. 2017] Aristidou A, Lasenby J, Chrysanthou Y & Shamir A (2017), 'Inverse kinematics techniques in computer graphics: A survey', *Computer Graphics Forum* . 31, 34, 36, 39, 149

[Atalay et al. 2017] Atalay A, Sanchez V, Atalay O, Vogt D. M, Haufe F, Wood R. J & Walsh C. J (2017), 'Batch fabrication of customizable siliconetextile composite capacitive strain sensors for human motion tracking', *Advanced Materials Technologies* 2(9). 22, 140, 149

[Ataollahi et al. 2016] Ataollahi A, Karim R, Fallah A. S, Rhode K, Razavi R, Seneviratne L. D, Schaeffter T & Althoefer K (2016), 'Three-degree-of-freedom mr-compatible multisegment cardiac catheter steering mechanism', *IEEE Transactions on Biomedical Engineering* 63(11). 111, 112, 145, 149

[Bächer et al. 2012] Bächer M, Bickel B, James D & Pfister H (2012), 'Fabricating articulated characters from skinned meshes', *ACM Transaction on Graphics (Proc. SIGGRAPH)* . 84, 150

[Bächer et al. 2016] Bächer M, Hepp B, Pece F, Kry P. G, Bickel B, Thomaszewski B & Hilliges O (2016), Defsense: Computational design of customized deformable input devices, *in* 'Proceedings of the CHI Conference on Human Factors in Computing Systems'. 22, 140, 150

[Back et al. 2016] Back J, Lindenroth L, Karim R, Althoefer K, Rhode K & Liu H (2016), New kinematic multi-section model for catheter contact force estimation and steering, *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems'. 110, 111, 150

[Baerlocher 2001] Baerlocher P (2001), Inverse kinematics techniques of the interactive posture control of articulated figures, PhD thesis, Ecole Polytechnique Federale de Lausanne. 34, 150

[Baerlocher & Boulic 1998] Baerlocher P & Boulic R (1998), Task-priority formulations for the kinematic control of highly redundant articulated structures, *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems.', Vol. 1. 34, 35, 150

[Bai et al. 2013] Bai L, Mitchell J & Pang J (2013), 'On convex quadratic programs with linear complementarity constraints', *Computational Optimization and Applications* . 98, 150

[Baraff & Witkin 1998] Baraff D & Witkin A (1998), Large steps in cloth simulation, *in* 'Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques'. 59, 150

[Barker et al. 2004] Barker J. L, Garden A. S, Ang K. K, O'Daniel J. C, Wang H, Morrison W. H, Rosenthal D. I, Chao K. C, Tucker S. L, Mohan R et al. (2004), 'Quantification of volumetric and geometric changes occurring during fractionated radiotherapy for head-and-neck cancer using an integrated ct/linear accelerator system', *Intl Jnl of Rad. Oncology Biology Physics* . 81, 82, 142, 150

[Bern et al. 2017] Bern J. M, Chang K.-H & Coros S (2017), 'Interactive design of

animated plushies', *ACM Transactions on Graphics* 36(4). 37, 150

[Braganza et al. 2007] Braganza D, Dawson D. M, Walker I. D & Nath N (2007), 'A neural network controller for continuum robots', *IEEE Transactions on Robotics* 23(6). 39, 41, 151

[Bretl et al. 2005] Bretl T, Lall S, Latombe J.-C & Rock S (2005), *Multi-Step Motion Planning for Free-Climbing Robots*. 136, 151

[Bryson & Rucker 2014] Bryson C. E & Rucker D. C (2014), Toward parallel continuum manipulators, *in* 'IEEE International Conference on Robotics and Automation (ICRA)'. 42, 151

[Burdick & Chirikjian 1998] Burdick J. W & Chirikjian G. S (1998), *The Kinematics of Hyper-Redundant Robots*. 13, 151

[Burgner-Kahrs et al. 2015] Burgner-Kahrs J, Rucker D. C & Choset H (2015), 'Continuum Robots for Medical Applications: A Survey', *IEEE Transactions on Robotics* 31(6). 18, 151

[Buss 2004] Buss S. R (2004), 'Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods', *IEEE Journal of Robotics and Automation* 17(1-19). 33, 34, 151

[Buss & Kim 2005] Buss S. R & Kim J.-S (2005), 'Selectively damped least squares for inverse kinematics', *Journal of Graphics Tools* 10(3). 34, 35, 151

[Calandra et al. 2015] Calandra R, Ivaldi S, Deisenroth M. P, Rueckert E & Peters J (2015), Learning inverse dynamics models with contacts, *in* 'IEEE International Conference on Robotics and Automation (ICRA)'. 45, 46, 151

[Caluwaerts et al. 2014] Caluwaerts K, Despraz J, Işçen A, Sabelhaus A. P, Bruce J, Schrauwen B & SunSpiral V (2014), 'Design and control of compliant tensegrity robots through simulation and hardware validation', *Journal of The Royal Society Interface* 11(98). 24, 151

[Chen & Goldfarb 2007] Chen L & Goldfarb D (2007), 'An active set method for

mathematical programs with linear complementarity constraints', *Computational Techniques and Applications* . 98, 99, 151

[Chen et al. 2014] Chen X, Zheng C, Xu W & Zhou K (2014), 'An asymptotic numerical method for inverse elastic shape design', *ACM Transaction on Graphics (Proc. SIGGRAPH)* . 84, 152

[Cheney et al. 2015] Cheney N, Bongard J & Lipson H (2015), Evolving soft robots in tight spaces, *in* 'Proceedings of the Conference on Genetic and Evolutionary Computation'. 25, 152

[Cheney et al. 2014] Cheney N, MacCurdy R, Clune J & Lipson H (2014), 'Unshackling evolution: Evolving soft robots with multiple materials and a powerful generative encoding', *SIGEVOlution* 7(1). 25, 152

[Cianchetti et al. 2015] Cianchetti M, Calisti M, Margheri L, Kuba M & Laschi C (2015), 'Bioinspired locomotion and grasping in water: the soft eight-arm octopus robot', *Bioinspiration Biomimetics* 10(3). 15, 152

[Cioranescu & Donato 1999] Cioranescu D & Donato P (1999), *An Introduction to Homogenization*, Vol. 17. 18, 152

[Coevoet, Escande & Duriez 2017] Coevoet E, Escande A & Duriez C (2017), 'Optimization-based inverse model of soft robots with contact handling', *IEEE Robotics and Automation Letters (Proc. ICRA)* 2(3). 28, 45, 46, 125, 137, 152

[Coevoet et al. 2018] Coevoet E, Escande A & Duriez C (2018), Soft robots locomotion and grasping control using fem simulation and quadratic programming. 28, 45, 46, 137, 152

[Coevoet, Morales-Bieze & et al. 2017] Coevoet E, Morales-Bieze T & et al. (2017), 'Software toolkit for modeling, simulation, and control of soft robots', *Advanced Robotics* 31(22). 27, 37, 38, 133, 137, 152

[Coevoet et al. 2014] Coevoet E, Reynaert N, Lartigau E, Schiappacasse L, Dequidt J & Duriez C (2014), Introducing interactive inverse fem simulation and its application for adaptive radiotherapy, *in* 'Medical Image Computing and Computer-Assisted Intervention'. 37, 38, 65, 69, 79, 81, 82, 138,

152

[Coevoet et al. 2015] Coevoet E, Reynaert N, Lartigau E, Schiappacasse L, Dequidt J & Duriez C (2015), 'Registration by interactive inverse simulation: application for adaptive radiotherapy', *International Journal of Computer Assisted Radiology and Surgery* 10(8). 37, 38, 69, 79, 82, 137, 153

[Cook et al. 2007] Cook R. D, Malkus D. S, Plesha M. E & Witt R. J (2007), *Concepts and Applications of Finite Element Analysis*. 23, 51, 53, 54, 153

[Coros et al. 2012] Coros S, Martin S, Thomaszewski B, Schumacher C, Sumner R & Gross M (2012), 'Deformable objects alive!', *ACM Transaction on Graphics (Proc. SIGGRAPH)* 31(4). 44, 46, 153

[Cotin et al. 2005] Cotin S, Duriez C, Lenoir J, Neumann P & Dawson S (2005), New approaches to catheter navigation for interventional radiology simulation, *in* J. S. Duncan & G. Gerig, eds, 'Medical Image Computing and Computer-Assisted Intervention'. 54, 153

[Courtecuisse et al. 2010] Courtecuisse H, Allard J, Duriez C & Cotin S (2010), 'Asynchronous preconditioners for efficient solving of non-linear deformations', *Virtual Reality Interaction and Physical Simulation, Eurographics Association* . 110, 129, 153

[Craig 2005] Craig J (2005), *Introduction to Robotics: Mechanics and Control*, Addison-Wesley series in electrical and computer engineering: control engineering, third edn. 31, 153

[Crum et al. 2004] Crum W. R, Hartkens T & Hill D. L. G (2004), 'Non-rigid image registration: theory and practice', *The British Journal of Radiology* 77. 81, 153

[Deo & Walker 1995] Deo A. S & Walker I. D (1995), 'Overview of damped least-squares methods for inverse kinematics of robot manipulators', *Journal of Intelligent and Robotic Systems* 14(1). 34, 153

[Dequidt et al. 2008] Dequidt J, Marchal M, Duriez C, Kerien E & Cotin S (2008),

Interactive simulation of embolization coils: Modeling and experimental validation, *in* D. Metaxas, L. Axel, G. Fichtinger & G. Székely, eds, 'Medical Image Computing and Computer-Assisted Intervention'. 54, 153

[Duindam et al. 2010] Duindam V, Xu J, Alterovitz R, Sastry S & Goldberg K (2010), 'Three-dimensional motion planning algorithms for steerable needles using inverse kinematics', *The International Journal of Robotics Research* 29(7). 32, 33, 154

[Dupont et al. 2010] Dupont P. E, Lock J, Itkowitz B & Butler E (2010), 'Design and control of concentric-tube robots', *IEEE Transactions on Robotics* 26(2). 110, 154

[Duriez 2013] Duriez C (2013), Control of elastic soft robots based on real-time finite element method, *in* 'IEEE International Conference on Robotics and Automation'. 37, 38, 69, 154

[Duriez et al. 2016] Duriez C, Coevoet E, Largilliere F, Morales-Bieze T, Zhang Z, Sanz-Lopez M, Carrez B, Marchal D, Goury O & Dequidt J (2016), Framework for online simulation of soft robots with optimization-based inverse model, *in* 'IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots'. 137, 154

[Duriez et al. 2006] Duriez C, Dubois F, Kheddar A & Andriot C (2006), 'Realistic haptic rendering of interacting deformable objects in virtual environments', *IEEE Transactions on Visualization and Computer Graphics* 12(1). 97, 154

[Erman & Mark 1997] Erman B & Mark J. E (1997), *Structures and properties of rubberlike networks*. 53, 154

[Escande et al. 2010] Escande A, Mansard N & Wieber P. B (2010), Fast resolution of hierarchized inverse kinematics with inequality constraints, *in* 'IEEE International Conference on Robotics and Automation'. 44, 46, 154

[Escande et al. 2014] Escande A, Mansard N & Wieber P.-B (2014), 'Hierarchical quadratic programming: Fast online humanoid-robot motion generation', *The International Journal of Robotics Research* 33(7). 44, 45, 46,

154

[Faure et al. 2012] Faure F, Duriez C, Delingette H, Allard J, Gilles B, Marchesseau S, Talbot H, Courtecuisse H, Bousquet G & Peterlik I (2012), Sofa: A multi-model framework for interactive physical simulation, *in* 'Studies in Mechanobiology Tissue Engineering and Biomaterials'. 25, 68, 155

[Felippa 2000] Felippa C. A (2000), A systematic approach to the element-independent corotational dynamics of finite elements, Technical report. 55, 155

[Ferreau et al. 2014] Ferreau H, Kirches C, Potschka A, Bock H & Diehl M (2014), 'qpOASES: A parametric active-set algorithm for quadratic programming', *Mathematical Programming Computation* 6(4). 69, 99, 155

[Ficuciello et al. 2018] Ficuciello F, Migliozzi A, Coevoet E, Petit A & Duriez C (2018), Dexterous manipulation for model-based deformation control of 3d soft objects, *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)'. 128, 137, 155

[Giallombardo & Ralph 2008] Giallombardo G & Ralph D (2008), 'Multiplier convergence in trust-region methods with application to convergence of decomposition methods for mpecs', *Mathematical Programming* . 99, 155

[Giorelli et al. 2012] Giorelli M, Renda F, Calisti M, Arienti A, Ferri G & Laschi C (2012), A two dimensional inverse kinetics model of a cable driven manipulator inspired by the octopus arm, *in* '2012 IEEE International Conference on Robotics and Automation'. 34, 35, 39, 155

[Giorelli et al. 2015] Giorelli M, Renda F, Calisti M, Arienti A, Ferri G & Laschi C (2015), 'Neural network and jacobian method for solving the inverse statics of a cable-driven soft arm with nonconstant curvature', *IEEE Transactions on Robotics* 31(4). 35, 39, 41, 155

[Giorelli et al. 2013] Giorelli M, Renda F, Ferri G & Laschi C (2013), A feed-forward neural network learning the inverse kinetics of a soft cable-driven manipulator moving in three-dimensional space, *in* 'IEEE International Conference on Intelligent Robots and Systems'. 39, 41, 155

[Golub & Van Loan 1996] Golub G & Van Loan C (1996), *Matrix computations*, 3rd edn, John Hopkins University Press. 101, 156

[Gould & Riggs 1986] Gould S. D & Riggs G. T (1986), 'Curving tip catheter'. US Patent 4,586,923. 111, 156

[Goury & Duriez 2018] Goury O & Duriez C (2018), 'Fast, generic and reliable control and simultation of soft robots using model order reduction', *IEEE Transactions on Robotics* . 24, 71, 72, 73, 87, 110, 141, 156

[Grochow et al. 2004] Grochow K, Martin S. L, Hertzmann A & Popović Z (2004), 'Style-based inverse kinematics', *ACM Transactions on Graphics* 23(3). 40, 41, 156

[Hannan & Walker 2003] Hannan M. W & Walker I. D (2003), 'Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots', *Journal of Robotic Systems* 20(2). 32, 33, 156

[Harish et al. 2016] Harish P, Mahmudi M, Callennec B. L & Boulic R (2016), 'Parallel inverse kinematics for multithreaded architectures', *ACM Transactions on Graphics* 35(2). 34, 35, 156

[Hauth & Strasser 2004] Hauth M & Strasser W (2004), Corotational simulation of deformable solids, *in* 'Winter School on Computer Graphics'. 24, 156

[Hawkes et al. 2017] Hawkes E. W, Blumenschein L. H, Greer J. D & Okamura A. M (2017), 'A soft robot that navigates its environment through growth', *Science Robotics* 2(8). 15, 121, 156

[Herzog et al. 2016] Herzog A, Rotella N, Mason S, Grimminger F, Schaal S & Righetti L (2016), 'Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid', *Autonomous Robots* 40(3). 44, 46, 156

[Hiller & Lipson 2014] Hiller J & Lipson H (2014), 'Dynamic simulation of soft multimaterial 3D-printed objects', *Soft Robotics* 1(1). 25, 156

[Holden et al. 2017] Holden D, Komura T & Saito J (2017), 'Phase-functioned neural

networks for character control', *ACM Transactions on Graphics* 36(4). 40, 41, 156

[Holden et al. 2015] Holden D, Saito J & Komura T (2015), Learning an inverse rig mapping for character animation, *in* 'Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation'. 40, 41, 157

[Holden et al. 2016] Holden D, Saito J & Komura T (2016), 'A deep learning framework for character motion synthesis and editing', *ACM Transactions on Graphics* 35(4). 40, 41, 157

[Hu et al. 2012] Hu J, Mitchell J, Pang J, Yu B, Hu J, Mitchell J, Pang J & Yu B (2012), 'On linear programs with linear complementarity constraints', *Journal of Global Optimization* . 98, 157

[Huang et al. 2001] Huang Q, Yokoi K, Kajita S, Kaneko K, Arai H, Koyachi N & Tanie K (2001), 'Planning walking patterns for a biped robot', *IEEE Transactions on Robotics and Automation* 17(3). 121, 157

[Hyun et al. 2014] Hyun D. J, Seok S, Lee J & Kim S (2014), 'High speed trot-running: Implementation of a hierarchical controller using proprioceptive impedance control on the mit cheetah', *The International Journal of Robotics Research* 33(11). 121, 157

[Ilievski et al. 2011] Ilievski F, Mazzeo A. D, Shepherd R. F, Chen X & Whitesides G. M (2011), 'Soft robotics for chemists', *Angewandte Chemie* 123(8). 125, 146, 157

[Ion et al. 2016] Ion A, Frohnhofen J, Wall L, Kovacs R, Alistar M, Lindsay J, Lopes P, Chen H.-T & Baudisch P (2016), Metamaterial mechanisms, *in* 'Proceedings of the 29th Annual Symposium on User Interface Software and Technology'. 17, 139, 157

[Jiang et al. 2017] Jiang H, Wang Z, Liu X, Chen X, Jin Y, You X & Chen X (2017), A two-level approach for solving the inverse kinematics of an extensible soft arm considering viscoelastic behavior, *in* '2017 IEEE International Conference on Robotics and Automation (ICRA)'. 39, 41, 157

[Johnson & Willemsen 2003] Johnson D. E & Willemsen P (2003), Six degree-of-freedom haptic rendering of complex polygonal models, *in* 'Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems'. 93, 158

[Jones & Walker 2006] Jones B. A & Walker I. D (2006), 'Kinematics for multisection continuum robots', *IEEE Transactions on Robotics* 22(1). 32, 33, 158

[Kikuchi & Oden 1988] Kikuchi N & Oden J. T (1988), *Contact problems in elasticity: a study of variational inequalities and finite element methods.* 94, 158

[Kim et al. 2008] Kim H. J, Wang Q, Rahmatalla S, Swan C. C, Arora J. S, Abdel-Malek K & Assouline J. G (2008), 'Dynamic motion planning of 3d human locomotion using gradient-based optimization', *Journal of biomechanical engineering* 130(3). 44, 46, 158

[Kim & Pollard 2011*a*] Kim J & Pollard N (2011*a*), 'Direct control of simulated non-human characters', *IEEE Computer Graphics and Applicaions* 31(4). 43, 44, 46, 158

[Kim & Pollard 2011*b*] Kim J & Pollard N (2011*b*), 'Fast simulation of skeleton-driven deformable body characters', *ACM Transaction on Graphics* . 44, 46, 158

[Kim et al. 2013] Kim S, Laschi C & Trimmer B (2013), 'Soft robotics: a bioinspired evolution in robotics', *Trends in Biotechnology* 31(5). 21, 158

[Koenig & Howard 2004] Koenig N & Howard A (2004), Design and use paradigms for gazebo, an open-source multi-robot simulator, *in* 'Proceeding of the conference on Intelligent Robots and Systems. (IROS). ', Vol. 3. 24, 26, 158

[Largilliere et al. 2015] Largilliere F, Verona V, Coevoet E, Sanz-Lopez M, Dequidt J & Duriez C (2015), Real-time control of soft-robots using asynchronous finite element modeling, *in* 'IEEE International Conference on Robotics and Automation'. 37, 38, 138, 158

[Lee et al. 2013] Lee D.-Y, Kim J.-S, Kim S.-R, Koh J.-S & Cho K.-J (2013), The

deformable wheel robot using magic-ball origami structure, *in* 'ASME international design engineering technical conferences and computers and information in engineering conference', American Society of Mechanical Engineers. 121, 158

[Li et al. 2017] Li S, Vogt D. M, Rus D & Wood R. J (2017), 'Fluid-driven origami-inspired artificial muscles', *Proceedings of the National Academy of Sciences* 114(50). 18, 19, 139, 159

[Liu et al. 2010] Liu L, Yin K, van de Panne M, Shao T & Xu W (2010), 'Sampling-based contact-rich motion control', *ACM Transactions on Graphics* 29(4). 44, 159

[Liu et al. 2013] Liu L, Yin K, Wang B & Guo B (2013), 'Simulation and control of skeleton-driven soft body characters', *ACM Transactions on Graphics* 32(6). 44, 46, 159

[Lo 2014] Lo D. S (2014), *Finite element mesh generation*, CRC Press. 71, 159

[Ma et al. 2017] Ma L, Zhang Y, Liu Y, Zhou K & Tong X (2017), 'Computational design and fabrication of soft pneumatic objects with desired deformations', *ACM Transactions on Graphics* 36(6). 16, 139, 159

[MacCurdy et al. 2016] MacCurdy R, Katzschmann R, Kim Y & Rus D (2016), Printable hydraulics: A method for fabricating robots by 3d co-printing solids and liquids, *in* 'IEEE International Conference on Robotics and Automation (ICRA)'. 16, 17, 18, 139, 159

[Majidi 2014] Majidi C (2014), 'Soft robotics: A perspective—current trends and prospects for the future', *Soft Robotics* 1. 12, 159

[Malomo et al. 2016] Malomo L, Pietroni N, Bickel B & Cignoni P (2016), 'Flexmolds: Automatic design of flexible shells for molding', *ACM Transactions on Graphics* 35(6). 16, 159

[Manti et al. 2015] Manti M, Hassan T, Passetti G, D'Elia N, Laschi C & Cianchetti M (2015), 'A bioinspired soft robotic gripper for adaptable and effective grasping', *Soft Robotics* 2(3). 73, 141, 159

[Martinelli 1997] Martinelli M. A (1997), 'Method and system for navigating a catheter probe'. US Patent 5,592,939. 111, 160

[Martínez et al. 2016] Martínez J, Dumas J & Lefebvre S (2016), 'Procedural Voronoi Foams for Additive Manufacturing', *ACM Transaction on Graphics* 35(12). 17, 139, 160

[Martins et al. 2006] Martins P. A. L. S, Natal J. R. M & Ferreira A. J. M (2006), 'A comparative study of several material models for prediction of hyperelastic properties: Application to silicone-rubber and soft tissues', *Strain* 42(3). 56, 160

[Melingui et al. 2015] Melingui A, Lakhal O, Daachi B, Mbede J. B & Merzouki R (2015), 'Adaptive neural network control of a compact bionic handling arm', *IEEE/ASME Transactions on Mechatronics* 20(6). 39, 41, 160

[Melingui et al. 2014] Melingui A, Merzouki R, Mbede J. B, Escande C, Daachi B & Benoudjit N (2014), Qualitative approach for inverse kinematic modeling of a compact bionic handling assistant trunk, *in* 'International Joint Conference on Neural Networks (IJCNN)'. 39, 41, 160

[Misra et al. 2008] Misra S, Ramesh K & Okamura A. M (2008), 'Modeling of tool-tissue interactions for computer-based surgical simulation: A literature review', *Presence: Teleoperators and Virtual Environments* 17(5). 56, 160

[Müller & Gross 2004] Müller M & Gross M (2004), Interactive virtual materials, *in* 'Proceedings of Graphics Interface'. 55, 160

[Murty 1972] Murty K. G (1972), 'On the number of solutions of the complementarity problem and spanning properties of complementarity cones', *Linear Algebra and its Applications* 5(1). 100, 160

[Muth et al. 2014] Muth J. T, Vogt D. M, Truby R. L, Mengüç Y, Kolesky B. D, Wood R. J & Lewis J. A (2014), 'Embedded 3d printing of strain sensors within highly stretchable elastomers', *Advanced Materials* 26(36). 22, 160

[Nakamura & Hanafusa 1986] Nakamura Y & Hanafusa H (1986), 'Inverse kinematic

solutions with singularity robustness for robot manipulator control', *Journal of dynamic systems, measurement, and control* 108(3). 34, 35, 160

[Nealen et al. 2006] Nealen A, Müller M, Keiser R, Boxerman E & Carlson M (2006), 'Physically based deformable models in computer graphics', *Computer Graphics Forum* 25(4). 23, 161

[Neppalli et al. 2009] Neppalli S, Csencsits M. A, Jones B. A & Walker I. D (2009), 'Closed-form inverse kinematics for continuum manipulators', *Advanced Robotics* 23(15). 32, 33, 161

[Nesme et al. 2009] Nesme M, Kry P. G, Jeřábková L & Faure F (2009), 'Preserving topology and elasticity for embedded deformable models', *ACM Transactions on Graphics.* 28(3). 18, 161

[Nguyen-Tuong & Peters 2011] Nguyen-Tuong D & Peters J (2011), 'Model learning for robot control: a survey', *Cognitive Processing* 12(4). 39, 40, 161

[Omidvar & Van der Smagt 1997] Omidvar O & Van der Smagt P (1997), *Neural systems for robotics*, Academic Press. 39, 161

[Panetta et al. 2015] Panetta J, Zhou Q, Malomo L, Pietroni N, Cignoni P & Zorin D (2015), 'Elastic Textures for Additive Fabrication', *ACM Transaction on Graphics* 34(10). 17, 161

[Paul 1981] Paul R. P (1981), *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*, Richard Paul. 31, 161

[Pérez et al. 2015] Pérez J, Thomaszewski B, Coros S, Bickel B, Canabal J, Sumner R & Otaduy M (2015), 'Design and fabrication of flexible rod meshes', *ACM Transaction on Graphics (Proc. SIGGRAPH)* 34(4). 84, 161

[Peterlik et al. 2011] Peterlik I, Nouicer M, Duriez C, Cotin S & Kheddar A (2011), 'Constraint-based haptic rendering of multirate compliant mechanisms', *IEEE Transactions on Haptics* 4(3). 26, 161

[Phaniteja et al. 2018] Phaniteja S, Dewangan P, Guhan P, Sarkar A & Krishna K. M (2018), 'A deep reinforcement learning approach for dynamically stable inverse kinematics of humanoid robots', *arXiv preprint arXiv:1801.10425* . 40, 41, 162

[Polydoros et al. 2015] Polydoros A. S, Nalpantidis L & Krüger V (2015), Real-time deep learning of robotic manipulator inverse dynamics, *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)'. 40, 41, 162

[Posa et al. 2013] Posa M, Cantu C & Tedrake R (2013), Direct trajectory optimization of rigid body dynamical systems through contact, *in* 'Algorithmic foundations of robotics X'. 44, 46, 162

[Rafsanjani et al. 2018] Rafsanjani A, Zhang Y, Liu B, Rubinstein S. M & Bertoldi K (2018), 'Kirigami skins make a simple soft actuator crawl', *Science Robotics* 3(15). 14, 15, 16, 19, 139, 162

[Raibert et al. 2008] Raibert M, Blankespoor K, Nelson G & Playter R (2008), 'Bigdog, the rough-terrain quadruped robot', *IFAC Proceedings Volumes* 41(2). 121, 162

[Reddy 1993] Reddy J. N (1993), *An introduction to the finite element method*, Vol. 2. 23, 54, 162

[Reddy 2013] Reddy J. N (2013), *An introduction to continuum mechanics*, Cambridge university press. 53, 162

[Rieffel et al. 2009] Rieffel J, Saunders F, Nadimpalli S, Zhou H, Hassoun S, Rife J & Trimmer B (2009), Evolving soft robotic locomotion in physx, *in* 'Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers'. 25, 162

[Robertson & Paik 2017] Robertson M. A & Paik J (2017), 'New soft robots really suck: Vacuum-powered systems empower diverse capabilities', *Science Robotics* 2(9). 19, 162

[Roche et al. 2017] Roche E. T, Horvath M. A, Wamala I, Alazmani A, Song S.-E,

Whyte W, Machaidze Z, Payne C. J, Weaver J. C, Fishbein G, Kuebler J, Vasilyev N. V, Mooney D. J, Pigula F. A & Walsh C. J (2017), 'Soft robotic sleeve supports heart function', *Science Translational Medicine* 9(373). 15, 162

[Roche et al. 2014] Roche E. T, Wohlfarth R, Overvelde J. T. B, Vasilyev N. V, Pigula F. A, Mooney D. J, Bertoldi K & Walsh C. J (2014), 'A bioinspired soft actuated material', *Advanced Materials* 26(8). 18, 19, 139, 163

[Rodríguez et al. 2017] Rodríguez A, Coevoet E & Duriez C (2017), Real-time simulation of hydraulic components for interactive control of soft robots, *in* 'IEEE International Conference on Robotics and Automation (ICRA)'. 37, 38, 64, 137, 163

[Rus & Tolley 2015] Rus D & Tolley M (2015), 'Design, fabrication and control of soft robots', *Nature* . 21, 163

[Santina et al. 2018] Santina C. D, Katzschmann R. K, B. A & Rus D (2018), Dynamic control of soft robots interacting with the environment, *in* 'IEEE International Conference on Soft Robotics'. 22, 87, 163

[Schreiber & Hirzinger 1998] Schreiber G & Hirzinger G (1998), *Singularity Consistent Inverse Kinematics by Enhancing the Jacobian Transpose*. 34, 35, 163

[Schumacher et al. 2015] Schumacher C, Bickel B, Rys J, Marschner S, Daraio C & Gross M (2015), 'Microstructures to control elasticity in 3d printing', *ACM Transaction on Graphics (Proc. SIGGRAPH)* 34(4). 17, 84, 163

[Sears & Dupont 2006] Sears P & Dupont P (2006), A steerable needle technology using curved concentric tubes, *in* '2006 IEEE/RSJ International Conference on Intelligent Robots and Systems'. 32, 33, 163

[Seok et al. 2013] Seok S, Onal C. D, Cho K. J, Wood R. J, Rus D & Kim S (2013), 'Meshworm: A peristaltic soft robot with antagonistic nickel titanium coil actuators', *IEEE/ASME Transactions on Mechatronics* 18(5). 18, 163

[Sha et al. 2002] Sha F, Saul L. K & Lee D. D (2002), Multiplicative updates for nonnegative quadratic programming in support vector machines, *in* 'Advances in neural information processing systems'. 70, 164

[Shepherd et al. 2011] Shepherd R. F, Ilievski F, Choi W, Morin S. A, Stokes A. A, Mazzeo A. D, Chen X, Wang M & Whitesides G. M (2011), 'Multigait soft robot', *Proceedings of the National Academy of Sciences* 108(51). 16, 164

[Siciliano 1990] Siciliano B (1990), 'Kinematic control of redundant robot manipulators: A tutorial', *Journal of Intelligent and Robotic Systems* 3(3). 13, 164

[Sifakis & Barbic 2012] Sifakis E & Barbic J (2012), Fem simulation of 3d deformable solids: A practitioner's guide to theory, discretization and model reduction, *in* 'ACM SIGGRAPH Courses'. 24, 51, 52, 53, 54, 56, 164

[Sifakis et al. 2005] Sifakis E, Neverov I & Fedkiw R (2005), 'Automatic determination of facial muscle activations from sparse motion capture marker data', *ACM Transactions on Graphics* 24(3). 36, 38, 164

[Sin et al. 2013] Sin F. S, Schroeder D & J. B (2013), 'Vega: Nonlinear fem deformable object simulator', *Computer Graphics Forum* 32(1). 55, 164

[Skouras et al. 2013] Skouras M, Thomaszewski B, Coros S, Bickel B & Gross M (2013), 'Computational design of actuated deformable characters', *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 32(4). 36, 164

[Steltz et al. 2010] Steltz E, Mozeika A, Rembisz J & N. Corson a. M. J (2010), Jamming as an enabling technology for soft robotics. 124, 145, 164

[Steltz et al. 2009] Steltz E, Mozeika A, Rodenberg N, Brown E & Jaeger H. M (2009), Jsel: Jamming skin enabled locomotion, *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems'. 18, 19, 121, 123, 124, 139, 145, 164

[Sueda et al. 2008] Sueda S, Kaufman A & Pai D. K (2008), 'Musculotendon simulation for hand animation', *ACM Transactions on Graphics* 27(3). 37,

38, 164

[Sugiyama & Hirai 2006] Sugiyama Y & Hirai S (2006), 'Crawling and jumping by a deformable robot', *The International Journal of Robotics Research* 25(5-6). 121, 122, 145, 165

[Suzumori 1996] Suzumori K (1996), 'Elastic materials producing compliant robots', *Robotics and Autonomous Systems* 18(1-2). 11, 12, 139, 165

[Suzumori et al. 1991] Suzumori K, Iikura S & Tanaka H (1991), Development of flexible microactuator and its applications to robotic mechanisms, *in* 'Proceedings. 1991 IEEE International Conference on Robotics and Automation'. 11, 165

[Talbot et al. 2012] Talbot H, Marchesseau S, Duriez C, Sermesant M, Cotin S & Delingette H (2012), 'Towards an interactive electromechanical model of the heart', *Interface Focus Royal Society* . 26, 165

[Talbot et al. 2015] Talbot H, Roy F & Cotin S (2015), Augmented reality for cryoablation procedures, *in* 'SIGGRAPH'. 26, 165

[Tan et al. 2012] Tan J, Turk G & Liu C. K (2012), 'Soft body locomotion', *ACM Transactions on Graphics* 31(4). 44, 46, 165

[Terryn et al. 2017] Terryn S, Brancart J, Lefeber D, Van Assche G & Vanderborght B (2017), 'Self-healing soft pneumatic robots', *Science Robotics* 2(9). 18, 19, 165

[Teschner et al. 2005] Teschner M, Kimmerle S, Heidelberger B, Zachmann G, Raghupathi L, Fuhrmann A, Cani M.-P, Faure F, Magnenat-Thalmann N, Strasser W & Volino P (2005), 'Collision detection for deformable objects', *Computer Graphics Forum* 24(1). 92, 165

[Tevatia & Schaal 2000] Tevatia G & Schaal S (2000), Inverse kinematics for humanoid robots, *in* 'Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)', Vol. 1. 35, 165

[Thieffry et al. 2017] Thieffry M, Kruszewski A, Goury O, Guerra T.-M & Duriez C (2017), Dynamic Control of Soft Robots, *in* 'IFAC World Congress'. 136, 166

[Thieffry et al. 2018] Thieffry M, Kruszewski A, Guerra T.-M & Duriez C (2018), Reduced order control of soft robots with guaranteed stability, *in* 'European Control Conference'. 87, 166

[Thomaszewski et al. 2014] Thomaszewski B, Coros S, Gauge G, Megaro V, Grinspun E & Gross M (2014), 'Computational design of linkage-based characters', *ACM Transaction on Graphics. (Proc. SIGGRAPH)* . 84, 166

[Thuruthel et al. 2017] Thuruthel G. T, Falotico E, Manti M, Pratesi A, Cianchetti M & Laschi C (2017), 'Learning closed loop kinematic controllers for continuum manipulators in unstructured environments', *Soft robotics* 4(3). 45, 46, 166

[Thuruthel et al. 2018] Thuruthel T. G, Ansari Y, Falotico E & Laschi C (2018), 'Control Strategies for Soft Robotic Manipulators: A Survey', *Soft Robotics* . 14, 31, 139, 166

[Thuruthel et al. 2016] Thuruthel T. G, Falotico E, Cianchetti M, Renda F & Laschi C (2016), Learning Global Inverse Statics Solution for a Redundant Soft Robot, *in* 'Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics', Vol. 2. 39, 41, 166

[Till et al. 2015] Till J, Bryson C. E, Chung S, Orekhov A & Rucker D. C (2015), Efficient computation of multiple coupled cosserat rod models for real-time simulation and control of parallel continuum manipulators, *in* 'IEEE International Conference on Robotics and Automation (ICRA)'. 42, 166

[Tolani & Badler 1996] Tolani D & Badler N. I (1996), 'Real-time inverse kinematics of the human arm', *Presence: Teleoperators & Virtual Environments* 5(4). 32, 33, 166

[Tolani et al. 2000] Tolani D, Goswami A & Badler N. I (2000), 'Real-time inverse kinematics techniques for anthropomorphic limbs', *Graphical Models*

62(5). 32, 33, 166

[Tolley et al. 2014] Tolley M. T, Shepherd R. F, Mosadegh B, Galloway K. C, Wehner M, Karpelson M, Wood R. J & Whitesides G. M (2014), 'A resilient, untethered soft robot', *Soft Robotics* . 18, 19, 121, 167

[Trivedi et al. 2008] Trivedi D, Rahn C, Kier W & Walkerc I (2008), 'Soft robotics: Biological inspiration, state of the art, and future research', *Applied Bionics and Biomechanics* . 12, 13, 14, 21, 139, 167

[Vikas et al. 2016] Vikas V, Cohen E, Grassi R, Sözer C & Trimmer B (2016), 'Design and locomotion control of a soft robot using friction manipulation and motor x2013;tendon actuation', *IEEE Transactions on Robotics* 32(4). 18, 19, 167

[Vogt et al. 2013] Vogt D. M, Park Y. L & Wood R. J (2013), 'Design and characterization of a soft multi-axis force sensor using embedded microfluidic channels', *IEEE Sensors Journal* 13(10). 22, 140, 167

[Wallin et al. 2018] Wallin T. J, Pikul J & Shepherd R. F (2018), '3d printing of soft robotic systems', *Nature Reviews Materials* . 16, 167

[Wang & Chen 1991] Wang L. C. T & Chen C. C (1991), 'A combined optimization method for solving the inverse kinematics problems of mechanical manipulators', *IEEE Transactions on Robotics and Automation* 7(4). 37, 38, 167

[Webster & Jones 2010] Webster R. J & Jones B. A (2010), 'Design and kinematic modeling of constant curvature continuum robots: A review', *International Journal of Robotics Research* 29(13). 32, 167

[Whitney 1969] Whitney D. E (1969), 'Resolved motion rate control of manipulators and human prostheses', *IEEE Transactions on Man-Machine Systems* 10(2). 33, 35, 167

[Wieber et al. 2016] Wieber P.-B, Tedrake R & Kuindersma S (2016), *Modeling and Control of Legged Robots*. 121, 167

[Yip & Camarillo 2014] Yip M. C & Camarillo D. B (2014), 'Model-less feedback control of continuum manipulators in constrained environments', *IEEE Transactions on Robotics* 30(4). 43, 46, 91, 143, 168

[Yip & Camarillo 2016] Yip M. C & Camarillo D. B (2016), 'Model-less hybrid position/force control: A minimalist approach for continuum manipulators in unknown, constrained environments', *IEEE Robotics and Automation Letters* 1(2). 43, 46, 168

[Yuen et al. 2014] Yuen M, Cherian A, Case J. C, Seipel J & Kramer R. K (2014), Conformable actuation and sensing with robotic fabric, *in* '2014 IEEE/RSJ International Conference on Intelligent Robots and Systems'. 16, 168

[Zehnder et al. 2017] Zehnder J, Knoop E, Bächer M & Thomaszewski B (2017), 'Metasilicone: Design and fabrication of composite silicone with desired mechanical properties', *ACM Transactions on Graphics* 36(6). 16, 139, 168

[Zhang et al. 2016] Zhang Z, Dequidt J, Kruszewski A, Largilliere F & Duriez C (2016), Kinematic modeling and observer based control of soft robot using real-time finite element method, *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems'. 22, 33, 35, 136, 168

[Zhang et al. 2017] Zhang Z, Morales Bieze T, Dequidt J, Kruszewski A & Duriez C (2017), Visual Servoing Control of Soft Robots based on Finite Element Model, *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems'. 22, 33, 35, 168

[Zhao & Badler 1994] Zhao J & Badler N. I (1994), 'Inverse kinematics positioning using nonlinear programming for highly articulated figures', *ACM Transactions on Graphics* 13(4). 36, 38, 168

[Zhu et al. 2017] Zhu B, Skouras M, Chen D & Matusik W (2017), 'Two-scale topology optimization with microstructures', *ACM Transactions on Graphics* 36. 17, 139, 168